Volume 39 Number 1 March 2015

ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

1977

Editorial Boards

Informatica is a journal primarily covering intelligent systems in the European computer science, informatics and cognitive community; scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor from the Editorial Board can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the list of referees. Each paper bears the name of the editor who appointed the referees. Each editor can propose new members for the Editorial Board or referees. Editors and referees inactive for a longer period can be automatically replaced. Changes in the Editorial Board are confirmed by the Executive Editors.

The coordination necessary is made through the Executive Editors who examine the reviews, sort the accepted articles and maintain appropriate international distribution. The Executive Board is appointed by the Society Informatika. Informatica is partially supported by the Slovenian Ministry of Higher Education, Science and Technology.

Each author is guaranteed to receive the reviews of his article. When accepted, publication in Informatica is guaranteed in less than one year after the Executive Editors receive the corrected version of the article.

Discriminating Between Closely Related Languages on Twitter

Nikola Ljubešić
University of Zagreb, Faculty of Humanities and Social Sciences, Ivana Lučića 3
E-mail: nikola.ljubesic@ffzg.hr, http://nlp.ffzg.hr/

Denis Kranjčić
University of Zagreb, Faculty of Humanities and Social Sciences, Ivana Lučića 3
E-mail: dkranjcic@ffzg.hr

Keywords: microblogging, language identification, closely related languages

Received: October 25, 2014

In this paper we tackle the problem of discriminating Twitter users by the language they tweet in, taking into account very similar South-Slavic languages – Bosnian, Croatian, Montenegrin and Serbian. We apply the supervised machine learning approach by annotating a subset of 500 users from an existing Twitter collection by the language the users primarily tweet in. We show that by using a simple bag-of-words model, univariate feature selection, 320 strongest features and a standard classifier, we reach user classification accuracy of ∼98%. Annotating the whole 63,160 users strong Twitter collection with the best performing classifier and visualizing it on a map via tweet geo-information, we produce a Twitter language map which clearly depicts the robustness of the classifier.

Povzetek: V prispevku raziščemo problem ločevanja uporabnikov družabnega omrežja Twitter glede na to, v katerem jeziku tvitajo, pri čemer obravnavamo zelo podobne južnoslovanske jezike: bosanščino, hrvaščino, srbščino in črnogorščino. Uporabimo pristop nadzorovanega strojnega učenja, kjer označimo vsakega uporabnika iz že obstoječe podatkovne množice 500 uporabnikov z jezikom, v katerem največ tvita. Pokažemo, da z uporabo enostavnega modela vreče besed, univariantno izbiro značilk, 320 najbolj pomembnih značilk in standardnim klasifikatorjem, dosežemo ∼97 % točnost klasifikacije posameznega uporabnika. Če uporabimo najboljši razviti klasifikator za označevanje naše celotne zbirke, ki zajema 63.160 uporabnikov, in rezultat prikažemo na zemljevidu z uporabo geografske informacija na tvitih, smo izdelali Twitter zemljevid jezikov, ki jasno pokaže robustnost razvitega pristopa.

1 Introduction

The problem of language identification, which was considered a solved task for some time now, has recently gained in popularity among researchers by identifying more complex subproblems, such as discriminating between language varieties (very similar languages and dialects), identifying languages in multi-language documents, code-switching (alternating between two or more languages) and identifying language in non-standard user-generated content which often tends to be very short (such as tweets).

In this paper we address the first and the last problem, namely discriminating between very similar languages in Twitter posts, with the relaxation that we do not identify language on the tweet level, but the user level.

The four languages we focus on here, namely Bosnian, Croatian, Montenegrin and Serbian, belong to the South Slavic group of languages and are all very similar to each other.

All the languages, except Montenegrin, use the same phonemic inventory, and they are all based on the write-as-you-speak principle. Croatian is slightly different in this respect, because it does not transcribe foreign words and proper nouns, as the others do. Moreover, due to the fairly recent standardization of Montenegrin, its additional phonemes are extremely rarely represented in writing, especially in informal usage. The Serbian language is the only one where both Ekavian and Ijekavian pronunciation and writing are standardized and widely used, while all the other languages use Ijekavian variants as a standard. The languages share a great deal of the same vocabulary, and some words differ only in a single phoneme / grapheme, because of phonological, morphological and etymological circumstances. There are some grammatical differences regarding phonology, morphology and syntax, but they are arguably scarce and they barely influence mutual intelligibility. The distinction between the four languages is based on the grounds of establishing a national identity, rather than on prominently different linguistic features.

2 Related work

One of the first studies incorporating similar languages in a language identification setting was that of [9] who, among others, discriminate between Spanish and Catalan with the accuracy of up to 99% by using second order character-level Markov models. In [11] a semi-supervised model is presented to distinguish between Indonesian and Malay by using frequency and rank of character trigrams derived from the most frequent words in each language, lists of exclusive words, and the format of numbers. [3] use a bag-of-words approach to classify Chinese texts from the mainland and Taiwan with results of up to 92% accuracy. [13] propose a log-likelihood estimation method along with Laplace smoothing to identify two varieties of Portuguese (Brazilian and European) obtaining 99.5% accuracy.

In the first attempt at discriminating between the two most distant out of the four languages of interest, namely Croatian and Serbian, [6] have shown that by using a second-order character Markov chain and a list of forbidden words, the two languages can be differentiated with a very high accuracy of $\sim$ 99%. As a follow-up, [12] add Bosnian to the language list showing that most off-the-shelf tools are in no way capable of solving this problem, while their approach by identifying blacklisted words reaches the accuracy of ~97%. [11] have worked with the same three languages as a subtask of producing web corpora of these languages. They have managed to outperform the best-performing classifier from [12] by training unigram language models on the entire content of the collected web corpora, decreasing the error related to the Croatian–Serbian language pair to a fourth. Recently, as a part of the DSL (Discriminating between Similar Languages) 2014 shared task of discriminating between six groups of similar languages on the sentence level [14], the language group A consisted of Bosnian, Croatian and Serbian and the best result in the group yielded 93.6% accuracy, which is not directly comparable to the aforementioned results because classification was performed on the sentence level, and not on the document level as in previous research.

Language identification on Twitter data has become a popular problem in recent years. [1] use language identification to create language specific Twitter collections of low-resource languages such as Nepali, Urdu, and Ukrainian. [2] use character n-gram distance with additional microblogging characteristics such as the language profile of a user, the content of an attached hyperlink, the language profile of mentioned users and the language profile of a hashtag. [7] review a wide range of off-the-shelf tools for Twitter language identification, and achieve their best results with a simple voting over three systems.

To the best of our knowledge, there has been only two attempts at discriminating between languages of high level of similarity on Twitter data. The first attempt dealt with Croatian and Serbian [4], where word unigram language models built from Croatian and Serbian web corpora were used in an attempt to divide users from a Twitter collection according to the two languages. An analysis of the annotation results showed that there is a substantial Twitter activity of Bosnian and Montenegrin speakers in the collection and that the the collected data cannot be described with a two-language classification schema, but rather with a 4-class schema that includes the remaining two languages. The second attempt focused on Spanish varieties spoken in five different countries [8] using geo-information as a gold standard, obtaining best results with a voting meta-classifier approach that combines the results of four single classifiers.

Our work builds on top of the research presented in [4] by defining a four-language classification schema, inside which Montenegrin, a language that gained official status in 2007, is present for the first time. Additionally, this is the first focused attempt at discriminating between those languages on Twitter data.

3 Dataset

The dataset we run our experiments on consists of tweets produced by 500 randomly picked users from the Twitter collection obtained with the TweetCat tool described in [4]. This Twitter collection consists currently of 63,160 users and 42,744,935 tweets. The collection procedure is still running which opens the possibility of the collection becoming a monitor corpus of user-generated content of the four languages.

For annotating the dataset there was only one annotator available. Annotating a portion of the dataset by multiple users and inspecting inter-annotator agreement is considered to be future work.

Having other languages in the dataset (mostly English) was tolerated as long as more than 50% of the text was written in the annotated language. Among the 500 users there were 10 users who did not comply to any of the four classes and were therefore removed from the dataset. One user, tweeting in Bosnian, had most of the tweets in English, there was one user tweeting in Macedonian and 8 users were tweeting in Serbian, but used the Cyrillic script. The users tweeting in Serbian and using the Cyrillic script were discarded from the dataset because we wanted to focus on discriminating between the four languages based on content and not the script used.

The result of the annotation procedure is summarized in the distribution of users according to their language, presented in Table 1. We can observe that Serbian makes up 77% of the dataset. There is a similar amount, around 9%, of Bosnian and Croatian data, while Montenegrin is least represented with around 5% of the data. These results are somewhat surprising because there is a much higher number of speakers of Croatian (around 5 million) than of Bosnian (around 2 million) or Montenegrin (below 1 million). Additionally, Croatia has the highest GDP of all the countries and one would expect that the adaptation rate of such new technology should be higher and not lower than in the remaining countries.

Because we plan to discriminate between the four languages on the user level, we are naturally interested in the amount of textual data we have at disposal for each in-

language	instance #	percentage
Bosnian (bs)	45	9.18%
Croatian (hr)	42	8.57%
Montenegrin (me)	25	5.10%
Serbian (sr)	378	77.14%

Table 1: Distribution of users by the language they tweet in.

stance, i.e. user. Figure 1 represents the amount of data available per user, measured in the number of words. The plotted distribution has the minimum at 561 words and the maximum at 29,246 words, whereas the arithmetic mean lies on 6,607 words. This distribution shows that we have quite a large amount of textual data available for the majority of users. We will inspect the impact of data available for predicting the language in Section 4.5.

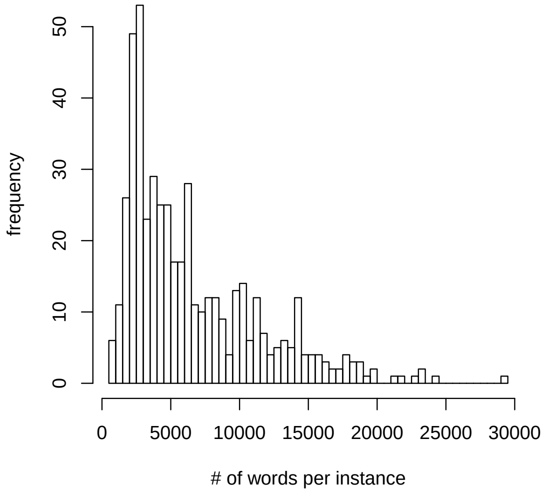

Figure 1: Distribution of dataset instances given the size in number of words.

4 Experiments

We perform data preprocessing, feature extraction and data formatting using simple Python scripts. All the machine learning experiments are carried out with scikit-learn [10]. Our evaluation metric, if not stated otherwise, is accuracy calculated via stratified 10-fold cross-validation.

We extract our features only from the text of the tweets. Using geolocation and user metadata (such as name, bio and location) is considered future work.

We experiment with the following preprocessing procedures:

- no preprocessing
- filtering out mentions, hashtags and URLs (making the data more representative of the user-generated content in general)
- dediacritizing the text (thereby lowering data sparsity)

and the following sets of features:

- words
- character 3-grams
- character 6-grams
- words and character 6-grams

Because no significant difference in accuracy was observed when using either different preprocessing procedures or sets of features (except for a slight drop when using character 3-grams), in the remainder of this section we present the results obtained by filtering out mentions, hashtags and URL-s and using words as features. By skipping dediacritization we keep the preprocessing level to a minimum, while by using words as features we ensure easy understandability of procedures such as feature selection. Finally, by removing textual specificities of Twitter like mentions and hashtags we ensure maximum applicability of the resulting models to other user-generated content besides tweets.

4.1 Initial experiment

The aim of the initial experiment was to get a feeling for the problem at hand by experimenting with various classifiers and features.

We experiment with traditional classifiers, such as the multinomial Naive Bayes (MultinomialNB), K-nearest neighbors (KNeighbors), decision tree (DecisionTree) and linear support-vector machine (LinearSVM). We use the linear SVM because the number of features is much greater than the number of instances. For each classifier we use the default hyperparameter values except for the linear SVM classifier for which we tune the C hyperparameter for highest accuracy.

classifier	accuracy ± stdev
DecisionTree	0.896 ± 0.026
KNeighbors	0.772 ± 0.040
LinearSVM	0.884 ± 0.034
MultinomialNB	0.806 ± 0.029

Table 2: Accuracy with standard deviation obtained with different classifiers using all words as features.

In the results presented in Table 2 we can observe that the LinearSVM and DecisionTree produce the highest accuracy. The significantly lower accuracy of the MultinomialNB classifier, which normally gives state-of-the-art results on bag-of-words models, but which has no inherent feature selection, provokes us to hypothesize that our results could improve if we applied explicit feature selection on our data. This follows our intuition that similar

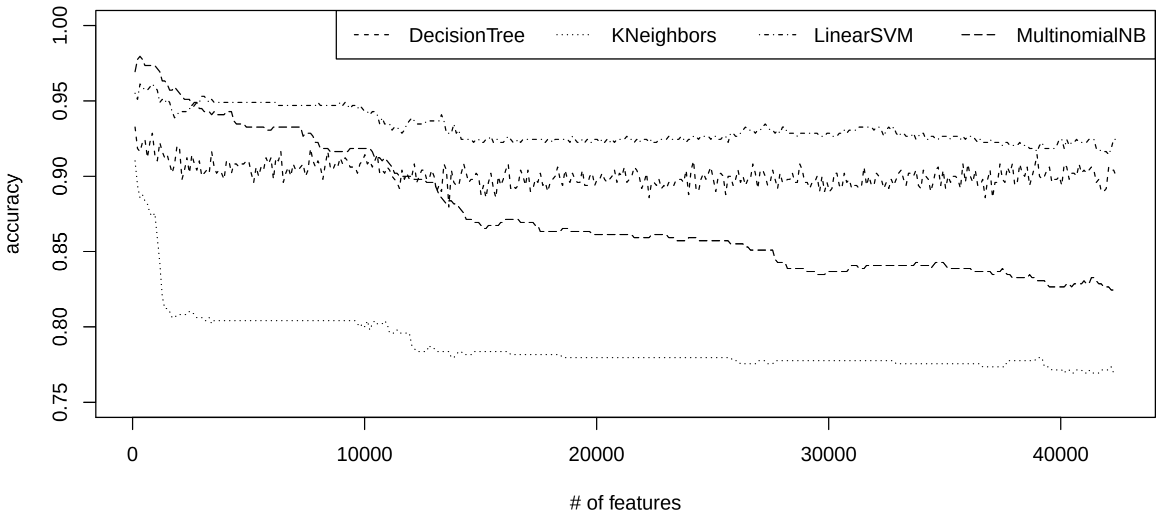

Figure 2: Classification accuracy as a function of number of most informative features used.

languages can be discriminated through a limited number of features, i.e. words, and not through the whole lexicon, which is normally shared to a great extent among such closely related languages.

4.2 Feature selection

Although there are stronger feature selection algorithms, we opt for a simple univariate feature selection algorithm which calculates p-value for each feature regarding the response variable through the F1 ANOVA statistical test. Finally it simply returns the user-specified number (or percentage) of features with lowest p-values. We use this simple feature selection method because we assume independence of our features, i.e. tokens or character n-grams, which is a reasonable assumption for language identification.

classifier	# of feats	acc ± stdev
DecisionTree	100	0.927 ± 0.019
KNeighbors	100	0.911 ± 0.041
LinearSVM	320	0.961 ± 0.025
MultinomialNB	320	**0.980 ± 0.016**

Table 3: Maximum accuracy obtained with each classifier with the number of strongest features used.

During these experiments we calculate accuracy via 10-fold cross-validation, performing feature selection each time on 90% of data used for model estimation.

The results of experimenting with up to 20% (cca. 42,000) of strongest word features are shown in Figure 2. Here we can observe a series of properties of the classifiers used. First of all, LinearSVM and DecisionTree, having implicit feature selection / weighting, operate similarly on the whole scale of number of features available, but still show better performance when using only a few hundred strongest features. On the other hand, MultinomialNB and KNeighbors show significantly better performance when they have to deal with the strongest features only. The best results are obtained with the MultinomialNB classifier at 320 features, reaching the accuracy of 97.97%. A numerical comparison of the best results obtained with the four classifiers is given in Table 3.

We present more detailed results obtained with the best-performing MultinomialNB classifier, trained on 320 features, in Table 4. It contains the confusion matrix of the classification process along with precision, recall and F1 obtained on each class. We can observe that the classification process is most successful on Serbian and Croatian, while the worst results are obtained on Montenegrin, which gets confused with both Bosnian and Serbian.

	bs	hr	me	sr	P	R	F1
bs	42	0	3	0	0.95	0.93	0.94
hr	1	41	0	0	0.98	0.98	0.98
me	0	0	23	2	0.82	0.92	0.87
sr	1	1	2	374	0.99	0.99	0.99

Table 4: Confusion matrix and precision, recall and F1 per class on the best performing classifier.

4.3 Evaluation on the test set

To perform a final test of our best performing classifier we produced an independent test set consisting of 101 annotated users. The MultinomialNB classifier, trained on all 490 users available from our development set, with 320 strongest features identified on that dataset, produces accuracy of 99.0%, having just one Bosnian user identified as Montenegrin. This experiment emphasizes the robustness of our classifier.

4.4 Analysis of the selected features

Using words as features, and not character 6-grams that perform equally well, enables us to easily interpret our final model. In Table 5 we present a systematization of the 320 features selected on the whole development set by language and the linguistic type of feature.

	bs	hr	me	sr
yat reflex	40.8	42.2	44.4	11.3
phonological	6.0	29.3	9.0	2.3
lexical	6.0	48.6	9.8	2.6
orthography	7.5	7.5	2.0	0.0
toponym, cultural	5.0	19.0	26.0	0.0
sum	65.3	146.8	91.3	16.2

Table 5: Feature type distribution across languages.

The features are divided into five categories across the four languages: yat reflex, phonological differences, lexical differences, orthography and toponym or cultural differences. Each feature contributes one point to the table: if a feature is present in more than one language, this point is divided among languages, and if a feature belongs to more than one feature type, the point is divided among those feature types. Almost half of the features belong to the "reflex of yat" category, which is least informative because most of the Ijekavian features are equally present in Croatian, Bosnian and Montenegrin. The exceptions are the words that are distinct both by the "reflex of yat" category and the lexical category, and few examples of Montenegrin-specific reflex of yat in words such as "nijesam" or "đe" (which also belongs to the "phonological differences" category). The "phonological differences" category contactins a lot of words present only in Croatian, such as "itko", "kava" or "večer" ("iko", "kafa" and "veče" in the other three languages). On the other hand, words that differ in only one phoneme and are not specific for Croatian are often spread among the remaining three languages. The category of lexical differences is similar in this respect: more than 70 percent of these features are Croatian. This can be explained by the fact that lexical purism is much more pronounced in Croatian than in the other three languages, which can be observed in the names of the months and some everyday words, such as "obitelj" (family), "glazba" (music), "izbornik" (menu) etc. In place of these words, Bosnian, Montenegrin and Serbian use words with evident foreign origin: "familija" (family), "muzika" (music), "meni" (menu) etc. The category of "orthography" predominantly contains infinitive verb forms without the final "i" letter, which appear in the future tense in Croatian orthography and which are also allowed in Bosnian. Finally, there is the category containing toponyms and culturally-specific items, such as country and city acronyms, names for residents, currency, TV-stations and even some public figures.

Although the features in the table are divided according to their real distribution among the languages, their distribution in the model sometimes differs. The reason for this is a significant difference between Croatian users and their language on the one side, and the rest on the other. Whereas Bosnian, Montenegrin and Serbian users are predominantly young people who use Twitter for chatting and sharing their everyday experiences, Croatian users are frequently news portals, shops, musicians, politicians etc. Consequentially, Croatian language on Twitter is marked by a much more formal register compared to the casual register of the other languages in our model.

4.5 Impact of amount of data available for prediction

Having a test set at our disposal opened the possibility of performing one additional experiment on the impact of the amount of data available for our language predictions. In our test set the user with the least amount of textual material contains 864 words. Therefore we evaluated the classifier trained on the whole development set by using only first N words from each user in the test set, N ranging from 10 to 850.

In Figure 3 we present the obtained results, representing each language with an F1 curve and all the languages with a micro-F1 curve. We can observe that the results peak and stabilize as we have 470 words at disposal for our prediction. This is an interesting result, showing that the large amount of data we have available for each user is actually not necessary. On the other hand, the results show quite clearly that discriminating between these languages on the level of each tweet would, at least with the presented classifier, be impossible given that the average tweet size is 10 words. Having significantly more training data available for each language could make a tweet-level classification possible since for Serbian, which covers 77% of the training data, on 10 words we already obtain a decent F1 of 0.88.

5 Corpus annotation and visualization

To be able to distribute separate Twitter collections of the four languages, we annotated each of the 63,160 users from our Twitter collection of Bosnian, Croatian, Montenegrin and Serbian. The annotation was performed with the MultinomialNB classifier trained on both the 490 development and the 101 testing instances, again selecting the 320 strongest features on that dataset.

Once we had our collection annotated, we decided to present the result of our language discriminator on a map.

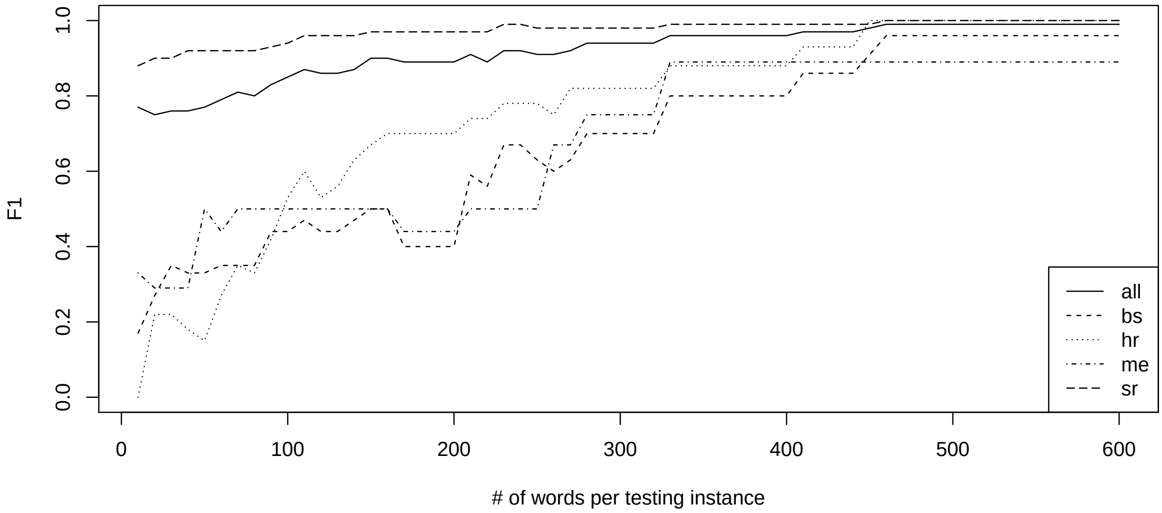

Figure 3: F1 per specific language as a function of the length of the testing instances.

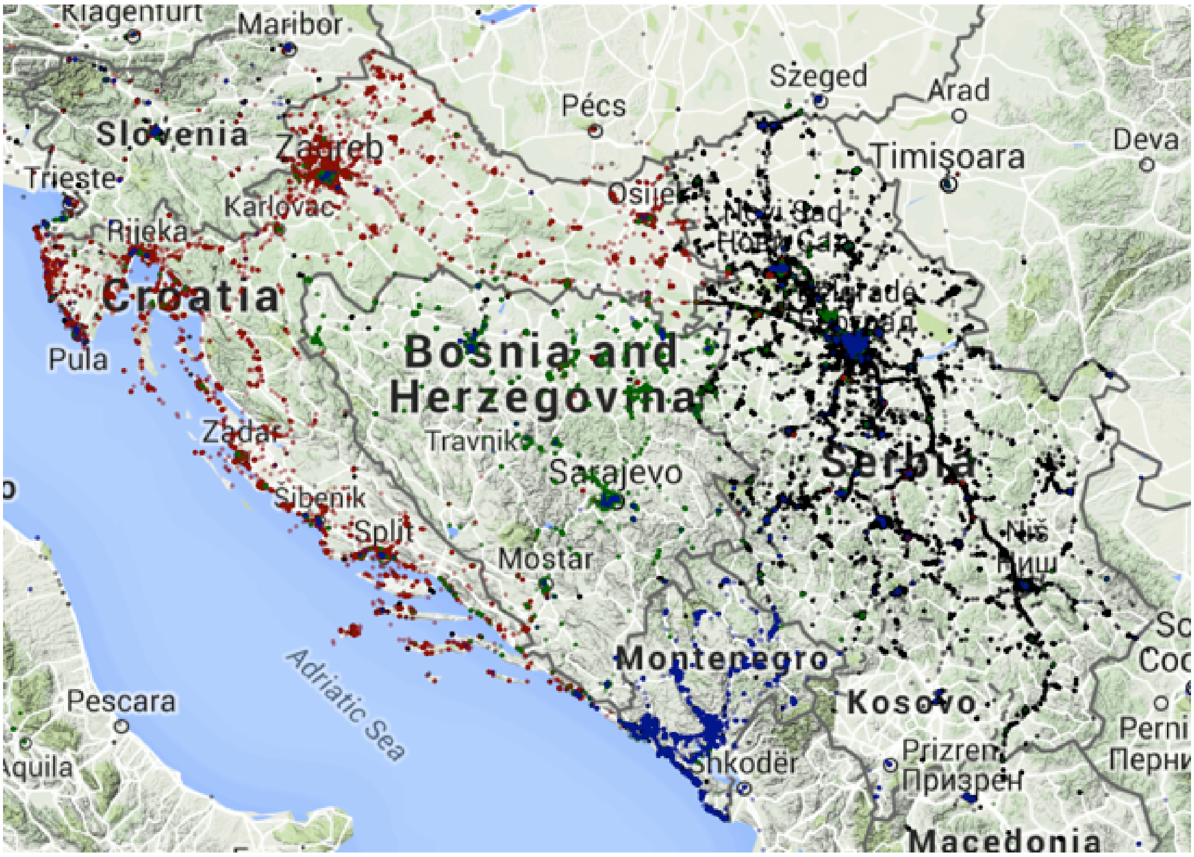

Figure 4: The Twitter language map. Impact of amount of data.

We presented each of the 576,786 tweets having geolocation available as a point on the map, encoding the predicted language of the author of the tweet with a corresponding color. We call the map presented in Figure 4 a “Twitter language map”.

The Figure shows the area of the four countries in which the four languages have official status. We can observe that the tweets follow quite consistently the country borders, which is an additional argument that our classifier works properly. From the plot we can also confirm that Twitter is much more popular and widespread in Serbia than in the remaining countries. Mixing of the four languages occurs, as one would expect, mostly in big cities, primarily Belgrade, the capital of Serbia. There we can observe a significant number of Montenegrin speaking Twitter users. To perform a sanity check regarding the correctness of these data, we manually inspected ten random users classified as being Montenegrin and tweeting in the wider Belgrade area. The inspection showed that all ten users actually tweet in Montenegrin.

Overall, we can observe that Croatia and Serbia have a higher amount of foreign-tweeting users which is easily explained by the well-known migrations from Bosnia to both Croatia and Serbia, and from Montenegro primarily to Serbia.

6 Conclusion

In this paper we have presented a straight-forward approach to discriminating between closely related languages of Twitter users by training a classifier on a dataset of 490 manually labeled users. By using the bag-of-words model, 320 strongest features regarding univariate feature selection and the multinomial Naive Bayes classifier, we obtained a very good accuracy of 97.97% on the development set and 99.0% on the test set. Best results were obtained on Croatian and Serbian while most errors occurred when identifying the Montenegrin language.

Analyzing the impact of data available for classification showed that classification accuracy stabilizes at $\sim$470 words per user which still does not enable us to use this classifier on the tweet level.

Finally we annotated the whole 63k-user-strong collection of tweets and presented the collection on a map we call the “Twitter language map”. The map shows that the language used on Twitter quite precisely follows the country borders, large cities being an exception to this rule.

Future work includes adding more information to our model besides words from tweets. Strongest candidates are the content to which users link and user meta-information such as username, location and bio. Using the geolocation information from tweets when available is surely a good source of information as well. Additionally, using the geolocation information as our response variable, i.e. redefining our task as predicting the location of a Twitter user is also a very interesting line of research. This surely increases the complexity of the task, but opens the door towards identifying dialects and sociolects.

Acknowledgement

The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement PIAP-GA-2012-324414 (Abu-MaTran).

References

[1] Bergsma, S., McNamee, P., Bagdouri, M., Fink, C., and Wilson, T. (2012). Language identification for creating language-specific twitter collections. In *Proceedings of the Second Workshop on Language in Social Media*, LSM ’12, pages 65–74, Stroudsburg, PA, USA. Association for Computational Linguistics.

[2] Carter, S., Weerkamp, W., and Tsagkias, M. (2013). Microblog language identification: overcoming the limitations of short, unedited and idiomatic text. *Language Resources and Evaluation*, 47(1):195–215.

[3] Huang, C.-R. and Lee, L.-H. (2008). Contrastive approach towards text source classification based on top-bag-of-word similarity. In *PACLIC*, pages 404–410. De La Salle University (DLSU), Manila, Philippines.

[4] Ljubešić, N., Fišer, D., and Erjavec, T. (2014). TweetCaT: a Tool for Building Twitter Corpora of Smaller Languages. In Chair), N. C. C., Choukri, K., Declerck, T., Loftsson, H., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC’14)*, Reykjavik, Iceland. European Language Resources Association (ELRA).

[11] Ljubešić, N. and Klubička, F. (2014). {bs,hr,sr}WaC – web corpora of Bosnian, Croatian and Serbian. In *Proceedings of the 9th Web as Corpus Workshop (WaC-9)*, pages 29–35, Gothenburg, Sweden. Association for Computational Linguistics.

[6] Ljubešić, N., Mikelić, N., and Boras, D. (2007). Language identification: How to distinguish similar languages. In Lužar-Stifter, V. and Hljuz Dobrić, V., editors, *Proceedings of the 29th International Conference on Information Technology Interfaces*, pages 541–546, Zagreb. SRCE University Computing Centre.

[7] Lui, M. and Baldwin, T. (2014). Accurate language identification of twitter messages. In *Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM)*, pages 17–25. Association for Computational Linguistics.

[8] Maier, W. and Gómez-Rodríguez, C. (2014). Language variety identification in spanish tweets. In *Proceedings*

of the EMNLP'2014 Workshop on Language Technology for Closely Related Languages and Language Variants, pages 25–35, Doha, Qatar. Association for Computational Linguistics.

[9] Padró, L. and Padró, M. (2004). Comparing methods for language identification. *Procesamiento del Lenguaje Natural*, 33:155–162.

[10] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[11] Ranaivo-Malancon, B. (2006). Automatic Identification of Close Languages – Case study: Malay and Indonesian. *ECTI Transactions on Computer and Information Technology*, 2(2):126–134.

[12] Tiedemann, J. and Ljubešić, N. (2012). Efficient discrimination between closely related languages. In *Proceedings of COLING 2012*, pages 2619–2634, Mumbai, India.

[13] Zampieri, M. and Gebre, B. G. (2012). Automatic identification of language varieties: The case of Portuguese. In *Proceedings of KONVENS2012 - The 11th Conference on Natural Language Processing*.

[14] Zampieri, M., Tan, L., Ljubešić, N., and Tiedemann, J. (2014). A Report on the DSL Shared Task 2014. In *Proceedings of the VARDIAL workshop*.

Call Routing Based on a Combination of the Construction-Integration Model and Latent Semantic Analysis: A Full System

Guillermo Jorge-Botana[1,3], Ricardo Olmos[2,3] and Alejandro Barroso[3]
[1]Universidad Nacional de Educación a Distancia (UNED), Calle Juan del Rosal, 10, Madrid, Spain.
[2]Universidad Autónoma de Madrid, Campus de Cantoblanco, 28049, Madrid, Spain.
[3]Semantia Lab. c/ Bravo Murillo nº 38, 28015-Madrid, Spain.
E-mail: gdejorge@psi.uned.es

Keywords: call routing, call steering, natural language, latent semantic analysis, construction-integration framework, psycholinguistically motivated algorithms

Received: June 24, 2014

This study stems from a previous article [1] in which we found that a psycholinguistically motivated mechanism based on the Construction-Integration (C-I) model [2,3] could be used for call classifiers in systems based on Latent Semantic Analysis (LSA). In it we showed that with this model more robust results were obtained when categorizing call transcriptions. However, this method was not tested in a context of calls in audio format, where a voice recognition application would be involved. The most direct implication of a voice recognition application is that the text to be categorized may be impoverished and is subject to noise. This impoverishment normally translates into deletions and insertions which are semantically arbitrary but phonetically similar. The aim of this study is to describe the behavior of a complete system, with calls in audio format that are transcribed by a voice recognition application using a Stochastic Language Model (SLM), and then categorized with an LSA model. This process optionally includes a mechanism based on the C-I model. In this study different parameters were analyzed to assess the automatic router's rate of correct choices. The results show that once again the model based on C-I is significantly better, but the benefits are more remarkable when the utterances are long. The paper describes the system and examines both the full results and the interactions in some scenarios. The economy of resources and flexibility of the system are also discussed.

Povzetek: Razvit je sistem za prepoznavanje govora z namenom uspešnega iskanja storitev ali entitet.

1 Introduction

Interactive Voice Response (henceforth, IVR) systems enjoy extremely widespread use today to provide customer service (Self Service). One of their drawbacks is that many of them consist of menus that limit user options to a few items, meaning that many intentions are not described within these items. This happens, for example, when a person wants something but believes that it is not represented in the menu. This constitutes a findability problem, since the category that represents what the person wants is not found. One of the alternatives that have been used is to employ spontaneous speech recognition techniques and subsequently categorize spontaneous utterances[1] into subject areas. These techniques are usually called “call routing” or “call steering” [4]. In them, rather than choosing between several items on a menu, the person simply hears an input such as “what would you like to do?” and responds spontaneously in natural language. What the person says is recognized with the help of an Automatic Speech Recognition (ASR) module, and once converted to text it is categorized and sent to a route where the user's needs are catered for. This is beneficial in terms of busy channels (the call will only be in one of the switchboard channels for a few seconds), and also beneficial in terms of convenience, as callers will be spared the effort associated with linking the representation of what they want to the representation of the categories expressed by the menu items, something which is not always intuitive [5, 6].

Generically speaking, the process of Call-Routing described above involves two steps. The first step, voice recognition, involves phonetic models of the language in which the service is offered, as well as a Stochastic Language Model (henceforth SLM), which is a formal representation of the probabilities of a word occurring if others have occurred beforehand. These SLM are habitually 3-gram or 4-gram models, frequently calculated using Maximum Likelihood Estimation and corrected by means of some Smoothing method (for example Good-Turing). In the end, the ASR module formulates its recognition hypotheses, taking the phonetic models of the language into account, as well as the probabilities provided by the SLM. The relative importance of each element (phonetic model vs. SLM) – in other words, the way of weighting the phonetic model over the SLM or vice-versa – can often be configured in voice recognition devices. Finally, from the scores

[1] Utterance: This is how we habitually refer to the phrase returned by the ASR module. We will also refer to the audio transcription of each call as an utterance.

yielded by the phonetic and SLM models, the ASR module generates a number of recognition hypotheses ordered by the confidence level which it assigns to each of them (a list of possible utterances). What the user has said is already in a text format, and now is the time to assign it to a destination - this will be the second step. To this end, classification techniques will be used to determine when a text (in this case the user request) belongs to one category or another (in this case to one destination or another).

Many classification techniques have been used to this end, with various results, depending also on the text sample and parameters [7]. Some examples are probabilistic models such as Maximum Entropy, artificial neural networks, and high-dimensional spaces, the category to which Latent Semantic Analysis (from now on LSA) would belong. All of the techniques mentioned have been extensively tested, but the current challenge is to achieve a more optimal and less biased representation of the words employed in the utterances, doing so even when data are not labeled or are retrieved and reanalyzed from samples which were not classified due to their difficulty [1, 8, 9].

In this study we have carried out a classification task employing LSA, but, as in these last studies, we have also opted to provide a better vectorial representation of the utterances. To this end, before performing the classification, we pre-processed the utterances using a technique based on a cognitive (or psycholinguistic) model that tries to account for the involvement of prior knowledge in the construction of meaning, and which also perfectly suits the philosophy underlying LSA: the Construction-Integration model [2, 3]. In fact, this study stems from a previous article [1] in which we found that a C-I based technique could be used for call classifiers, but we wanted to test it in a case which involved a voice recognition application, in order to show the performance of the entire system. Both LSA and C-I will be described in more detail in later sections.

2 LSA and Call Routing

LSA was originally described as a method for Information Retrieval [10], although some authors went beyond the original conception and adapted it as a psychological model of the acquisition and representation of lexical knowledge. In recent years its capacity to simulate aspects of human semantics has been widely demonstrated. For example, it adequately reflects why children learn the meanings of words without the need for massive exposure [11]. It is, to summarize, a technique derived from the field of application, but which models some parts of the human linguistic behavior. This in turn can provide benefits for the field of application.

LSA is a vectorial representation of language which is constituted by exploiting word occurrences in different contexts using large linguistic corpora (or text samples). It can be conceived as an automatic sequence of mathematical and computational methods, including pruning, weighting and dimension reduction via Singular Value Decomposition (SVD), to represent the meaning of words and text passages [10, 11]. A key issue is that every term or document (a document is a paragraph or sentence, or in the case of call categorization, a destination cluster or a call) in the corpus is expressed as a k-dimensional vector. Once the whole process is carried out, the cosine of the angle between two vectors is frequently used to evaluate the semantic relationship between the two terms or between the two documents corresponding to the vectors (formula 1). A close semantic relationship between two words, or between two documents, is shown by a high cosine, with a high value that is close to one, whilst two semantically unrelated words or documents have a cosine that is close to zero (orthogonally) or even a slightly negative one. In addition, sentences or texts that are not in the document matrix can be introduced as if they actually were included, using a technique commonly known as Folding-in, which projects this new document into the document matrix (formula 2). In the case of categorization, a vector of the user's utterance will be projected, plus one vector for each of the exemplar texts that represent the destinations, in such a way that if they are similar it will be inferred that what the user wants must be routed to this destination. A new vector d can be created by computing an utterance c (a new vector column in the occurrence matrix including all the terms that occur in it) and then multiplying it by the term matrix, usually called U, and the inverse of the diagonal matrix, usually called S; c is also computed by applying the same weights as in the creation of the original space.

$$Cos(V_1, V_2) = \frac{V_1 \cdot V_2}{|V_1||V_2|} \quad \text{(1) Similarity}$$

$$d = c^T U S^{-1} \quad \text{(2) Folding-in}$$

In the field of Call Routing, LSA has been used mainly with two motivations: correction of speech recognition hypotheses and assignation of utterances to destinations (both are occasionally used in the same router). Concerning the correction of speech recognition output, some studies have tested lists of common potential confusions on the part of speech recognition applications (homophones and near-homophones, {affect, effect}, {quiet, quite}), obtaining good results if such mistakes were corrected by checking the contextual coherence with indices of semantic similarity taken from an LSA model [12]. Satisfactory results were obtained if these confusions were corrected by checking the contextual coherence by means of indices of semantic similarity from an LSA model. These results were better than those of a model that combines a trigram model of the parts of speech with a Bayesian classifier, leading to the conclusion that LSA is a good option in these conditions.

Another study confirmed the usefulness of LSA in conjunction with a model based on n-grams [13]. This study concludes that the analysis by means of n-grams is limited to a text window that is too local (normally 3 or 4 words), and that the LSA model greatly enriches the results from the router, if the probabilities of the predicted word due some history sequence are also calculated in terms of the LSA model, that is, by means of the similarity between the word and the text of the history, which is projected as a pseudodocument. This probability, based on the Semantic Space, will be highest for words whose meaning aligns most closely with the meaning of the history and lowest for words that do not convey any particular information about it, for example, function terms. It is argued that this fact is complementary to n-grams models, because it is its exact opposite. n-gram models tend to assign higher probabilities to (frequent) function words than to (rarer) content words. So the added value of this study is to integrate both kind of probabilities in the generic LM (Language Model) equations.

Some authors extend this same procedure, improving the representation of the history sequence that acted as a pseudodocument in the original study [14]. Due to the briefness of the history for the utterances (and hence its pseudodocument), they replace the history pseudodocument by a more representative one, extracted from its semantic vicinity. They also use the first nearest semantic neighbors to estimate unseen sequences, as is usually done when using smoothing methods.

Again, other studies also obtained good results by interpolating the LSA model and the n-gram model [15], directly correcting speech recognition hypotheses and reassigning new probabilities to them, taking into account the coherence of the lexical context that accompanies each word [16]. Similarly, a new study performed a series of experiments to correct classification errors [17]. They corrected speech recognition outputs using indices of syntactic and semantic coherence (and a combination of both). As for LSA, the authors reported that it yielded results comparable to those of Word-Net [18, 19] (LSA being more economical and flexible), and that, when certain parameters were used, a combination of the two measures led to an improvement in error correction. This same philosophy has also motivated some studies in which the probabilities of the sequences of SLMs - in other words, the likelihood of a word occurring given a history h (sequence of words that precede it) – are recalculated on the basis of the semantic similarity between the word vector and the vector of its history [20].

As for assigning utterances to destinations, a first study proposed a system in which the user response to the typical “say anything” cue is classified by an LSA module according to candidate destinations [4]. The module will compare the vector representation of what the user has said (utterance vector) with the vectors that represent each of the destinations, made up of a compilation of all the calls that belong to each of those destinations. This module also has a disambiguation mechanism in the event that the utterance vector is similar to several destinations. In this case, terms will be found that represent the difference vectors between the utterance vector and each of the destination vectors. Once found, only the terms that may form bigrams or trigrams with a term from the original request will be used. These terms are used to formulate questions for the users in order to disambiguate. One peculiarity of this study is that they used 4,497 transcriptions from a banking services call-center in the LSA training phase. The occurrence matrix to begin the LSA process consists of terms and routes (rather than terms and transcriptions), and as a result few columns are produced – 23, to be precise. This is precisely the criticism made in a later study [21]: the authors specified that they took the possible destinations rather than call transcriptions as documents, so that the LSA training was quite limited. They obtained better results in their laboratory if the documents of the matrix were composed of call transcriptions. Another study introduced a variant in the preprocessing stage [22]. When the corpus was trained, the Information Gain (IG) was calculated in order to identify the terms that actually contributed useful information to the router. The IG index is based on the variations in document entropy (the amount of information carried by a document) with and without the term analyzed. Good results were later also reported when using a training variant in which the labels flagging the transcriptions (the destinations or routes) were entered as terms [23]. This enabled the authors to bring together transcriptions that had been routed to the same places. Other authors have also obtained improvements by introducing an additional step between recognition and Call Routing [24]. These authors did not enter the “utterance” as collected by the ASR module (with a generated SLM) directly –– rather they corrected it by using the LSA model confidence indices. Using this method, calls that were more than eight words long (about 12 words) were routed slightly better, and this improvement is greater if the ASR module has laxer criteria (a lower confidence threshold of acceptance).

As for study [24], the motivation for this study is also to introduce an additional step between recognition and LSA based Call Routing in order to differentially reassign the importance of each term of each utterance. This is done by means of the Construction-Integration model [3,2] (henceforth, C-I). C-I is a psycholinguistically motivated model that seeks to simulate working memory and real-time comprehension mechanisms, but it can be applied to categorization tasks as in the case of Call Routing. We will return to this model in a later section.

3 Objectives

There are two main objectives of this study, a general one and a specific one:

1. To implement (and describe in detail) a real full system in which we used LSA and a mechanism based on the Construction-Integration model (C-I).
2. To study the efficiency of the system in a real speech recognition situation by analyzing the percentage of correct classified utterances using different parameters. The percentage of correct classified utterances, thus, showed the quality of the system.

The manipulated parameters were (1) the use or not of an additional step based in C-I (over a LSA-based Call Routing), (2) the length of utterances, (3) the decrease in recognizer performance, and (4) the number of hypotheses processed from the n-best list derived from the voice recognition application.

4 Functional description of the system

One of the aims of this paper is to present a detailed study of the results obtained in a real call router in which a mechanism based on Kintsch's Construction-Integration model [2] was introduced. This model had already been partially tested in a previous study [1], but in this study the full process is described, including voice recognition for spontaneous speech and classification according to possible destinations. The system comprises several modules and is domain-independent, as it was trained using texts from different subject areas. For this study, the domain subject area was a customer service at a Telco (credit balance, top-ups, complaints, phones, etc.)

4.1 The voice recognition phase

Prior to any semantic interpretation, we need a voice recognition stage. The process consists of several layers. The first layer is responsible for recognizing words using the phonetic model of a language. There are numerous packages that perform voice recognition using a phonetic model of a particular language, based on sequences of letters and their pronunciation. These packages usually have standard dictionaries that specify the pronunciation of very common words, as well as dictionaries for the pronunciations that the integrator itself considers to be correct. In general, for recognition of phrases to occur, we need to explicitly specify which phrases we expect will be uttered. They are specified using deterministic grammars in different formats (abnf, grxml, gsl, etc.).

But if our aim is the recognition of spontaneous speech, we must generate a statistical language model (SLM), which uses a large linguistic corpus to generate a model where the probabilities of some words appearing are specified, given those that have occurred previously. To calculate these probabilities, Maximum Likelihood Estimation (from now on ML) is commonly used, corrected by a smoothing method that estimates the occurrences of words within some ranges [7]. One of these smoothing methods is Good-Turing. The package that we use to calculate probabilities in our model is SRILM [25]. It has Good-Turing as the default method (see http://www.speech.sri.com/projects/srilm/manpages/ngram-discount.7.html). It works as follows: by default, the unigrams that occur more than once and the n-grams that occur more than seven times are considered reliable. For this reason, standard ML is applied to calculate probabilities. But if the n-grams occur less than seven times, a correction is applied to the probability extracted from ML using the Good-Turing smoothing technique. It is also possible to estimate n-grams that do not occur in the reference corpus with the Katz method, using the BOW (back-off weight) of the history of each n-gram and the smoothed probabilities, but for simplicity's sake, this is not implemented in our system. Therefore, our model only contains conditional probabilities of n-grams that appear in the training corpus that have been smoothed (using the Good-Turing method).

As was previously mentioned , all these calculations are carried out by the SRILM package, in which individual scripts are used to program SLM generation - in our case with classes that group words (days of the week, months, countries, cities, etc.). In the end, the complete model is output into a file with the .arpa extension (Advanced Research Projects Agency format), whose main use is for exchanging language models. Using this .arpa file, we generate a .grxml file where the corrected probabilities from the input file are recorded several times in the form of a tree, but this time in SGRS (Speech Grammar Recognition Specification) format, which can be read by many commercially available recognition packages. In our case, we will use the grxml of the generated SLM in a Nuance 9 recognition engine.

4.2 The call-routing phase

Once the recognition hypotheses were established - in other words, what was recognized by the recognition engine- we proceeded with the second process: categorization by destination. In the case of this study, as noted earlier, the categorization procedure is implemented using LSA. So we needed a LSA semantic space to project the user's utterance as well as each of the exemplar texts that represent the destinations in it.

4.2.1 Semantic Space

To obtain that semantic space, we used a training corpus of digitized utterances belonging to several phone companies. The LSA was trained and the semantic space was created using this corpus. These utterances were extracted using the Wizard of Oz procedure (a technique where the users are made to believe that they are interacting with a computer but in fact they interact with a person) and the transcriber labeled each of them with the destination to which it was routed. We should note that the labeling was performed using different criteria, was carried out at different times and at different companies, and is not exhaustive. In any case, to provide cohesion for interrelated words, these labels were regarded as additional words, as in Featured LSA [23]. In the end, the LSA comprises the terms and labels that occur in utterances. In a previous study it was demonstrated that retaining these labels boosts categorization performance and produces a positive interaction with usage of C-I, even if they are non-exhaustive [1]. It should also be borne in mind that the only utterances that must be exhaustively labeled are those that are part of the destination sample (see section 4.2).

In the LSA training we used Gallito® (see www.elsemantico.es), a tool that has been used in other occasions for the creation of semantic spaces [26, 27, 28]. The words matching a special stop-list for this domain were eliminated, as were all function words. We also eliminated words that did not occur at least three times. Some words which are relevant within the telephony corpus were artificially grouped into a single class, for example countries, mobile phone brands or telecommunications companies, substituting them with the name of the class. In the end, we obtained a matrix of 1,421 terms and 34,905 utterances. In a next step, log-Entropy is calculated in this matrix. log-Entropy estimates the amount of information that a word carries in the documents where it appears. In this way, the terms that might contain the most information are given a heavier weight. So from this last calculation we got a weighed matrix to which SVD is applied and the three resulting matrices are reduced to 270 dimensions. We chose such a dimensionalization based on the assumptions made in previous studies [28]. In those studies it was suggested that the optimal number of dimensions for specific domain corpora does not have to be extremely low, sometimes even approaching the 300 dimensions recommended by Landauer, et al. [11]. In summary, the result of the entire process, the three reduced matrices (terms, diagonal and utterances matrix), is the semantic space of the mobile telephony domain that will be used as a basis for projecting the user's utterance as well as texts that represent the destinations.

4.2.2 Destination Management

The service we wish to evaluate has 29 basic destinations, covering the needs of a telephone company call center. Previous LSA-based Call Routing research compared the vector representation of what the user says (what the ASR module returns) in real time with each of the utterances used in training that are labeled with a destination. After the comparison is performed, the label of the most similar exemplar is selected [4]. This label will be the destination selected. The philosophy of our system is similar, although three important points should be highlighted:

The first one is that in our system we do not use all utterances from the training corpus as destination exemplars, but rather focus on a representative sample. In particular, a total of 2,329 calls are part of this sample (which are loaded and transformed into a vector of the semantic space only once the router has been launched). This is important because it is not necessary for all the calls from the training corpus to be labeled and participate as exemplars. We manually selected only a few representative calls (an excerpt is shown in Table 1), making the process more economical and the response to changes in the routing model (a change in the definition of the call-center destinations) faster. This is a very important issue if someone wants a successive deployment of a Call Routing service [29]. Massive annotation used to be very time-demanding. We made this pre-selection very carefully given that, as was pointed out in some studies [8], system performance depends on the quality of these destination exemplars.

1	ALTASBAJASC	well I'd like to cancel this phone number
1	ALTASBAJASC	switch to a monthly contract
1	ALTASBAJASC	to cancel a monthly contract
1	ALTASBAJASC	activate a number
1	ALTASBAJASC	cancel the mobile
1	ALTASBAJASC	I'd like to get a phone with you

Table 1: Excerpt from the destination exemplars.

The second point is that to decide what the most credible destination or destinations are we do not simply individually compare the user utterance with each of the utterances in the destination sample. In this system we use a method called average-4 which proved to be better in a previous study [1]. This method averages the four exemplars with the greatest cosine for each destination. The chosen destination is the one where the average of the four exemplars is the highest. In this way, any bias which an anomalous exemplar (seemingly very similar to what the user said) might have is eliminated.

The third point is that after converting the user utterance into a vector and selecting the most likely destinations, a list of the first four destinations is returned in descending order of their cosines. When the user is not sent toward the first option, this allows the first destination and the next most likely destinations to be used as candidates, disambiguating with an explicit question. This will depend on the design of the dialogs.

What is common to earlier studies is that both the destination exemplars and the user utterance will be converted into vectors that can be interpreted by the LSA space (in other words, use of the semantic vector space will be essential). The way the destinations and user utterances are converted into vectors will depend on the effectiveness of the system. In the following section we explain different ways of doing so.

4.2.3 Construction-Integration Model vs. direct method

There are two ways of representing each of the utterances vectorially in the router, whether utterances are destination exemplars or user utterances. One is Direct routing, where the utterances are projected onto the latent semantic space without any kind of additional algorithm. This is the standard LSA method for constructing new documents in the vector space, and is known as Folding-In [10]. The second is C-I routing, which, in contrast to Direct Routing, has an intermediate step in the construction of both destination exemplars and user utterances. This intermediate step is based on a Construction-Integration network (Figure 1). The importance that a Construction-Integration network might have for routers lies in the fact that the words in an utterance are modulated to their correct meaning, taking into account the entire lexical context, which constrains the final meaning. This is particularly relevant for words that are ambiguous (such as “card”) - the meaning that best matches the context will be given priority, thus avoiding predominant sense inundation and other biases frequently observed in vector space models [26, 27]

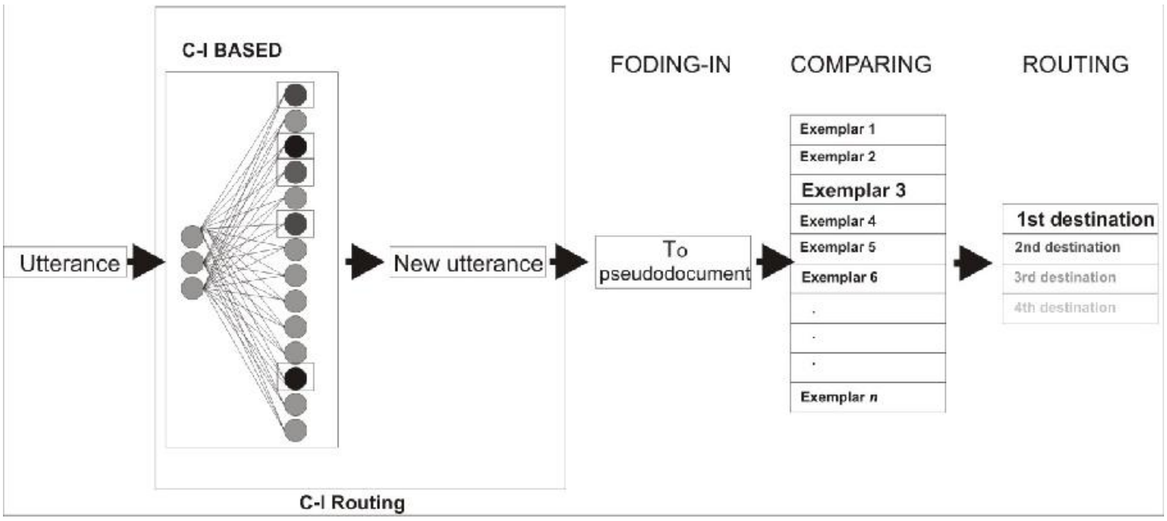

Figure 1: Graphical view of the Call Routing process for Direct routing and C-I routing.

Computational models based on Construction-Integration [2, 3] are based on the idea that the meaning of a text is constructed and integrated in the working memory. The mechanism retrieves the content linked to every word of a text from long-term memory, and, once retrieved, ensures that all this content is integrated in real time using a mutual constraint mechanism in the working memory. Therefore, it is a model that seeks to simulate working memory and real-time comprehension mechanisms [30]. Looking at the model details, interpretation of a text is carried out in two phases. In the first phase, all terms related to each of the words that make up the text or phrase are evoked. A network is constructed from all of them, where each one is joined to all the others - this phase is known as the Construction phase. In the second phase, known as the integration phase, each of these terms receives activation from the others proportionally with respect to the similarity they have with each other. After this phase, the most activated terms are those related to the main idea in the text [3,30]. This takes into account the fact that a text is not the sum of the terms included in it, but rather the integration of all of them into a final idea. Each term is constrained by the meaning of the other terms, thus generating the meaning in real time. This type of mechanism has been used on predicate and metaphorical structures [31, 32, 33] and on structures enriched with syntactic dependency relationships, thus demonstrating that the meaning of complex phrases can be modeled [34]. In operational terms, this mechanism makes it possible to differentially reassign the importance of each term. Ultimately, it is an estimate of the relevance of the utterance terms, as was done in previous studies [8].

How is this model implemented in our system? For each utterance, whether it is a user utterance or destination exemplars, a network is constructed based on the Construction-Integration network whose launch will lead to the extraction of new terms (see Figure 2). We might say that these terms produce a better definition of the utterance, an idea common to all the words contained in the original utterance. The procedure, in terms of its functionality, is analogous to that used by a study [35] implemented to improve classical methods for evaluating text with LSA. Figure 2 provides a graphic description of the procedure. Firstly (in the Construction phase), each term of the utterance is compared to all of the terms present in the semantic space, and the 200 neighbors most similar to each of them are extracted (this similarity is calculated using the cosine). A connectionist network is created between all these neighbors (neighbor node layer) and each of the original terms of the utterance (utterance node layer), where the weight of each connection is given by the cosines[2] between each of the connected terms (figure 2). Once the weights of the connections have been assigned, the activation of each node is calculated based on the connections received, in the second Integration phase (see formula 3). Thus, the greater the weight of the connections received, the greater the activation. The activation function also favors instances where the source of activation derives from several terms in the utterance, and not just one or two (due to the parameter δ in formula 3).

$$A_i = \text{¢}_i \sum_{j=1}^{n} \log(1 + C_{ij}) \quad \text{(3) Activation function}$$

Where *j* is the sub-index of the utterance layer, *i* is the sub-index of the neighbors layer, Cij is the strength of the connection that node *i* received from node *j* (the latter node belonging to the first layer), and *δ* is a correction factor to avoid unilateral activation (based on the standard deviation of the connections received), as defined in formula 4:

$$\text{¢}_i = \frac{1}{\sqrt{\frac{\sum_{j=1}^{n}(C_{ij} - \overline{C}_i)^2}{n} + 1}} \quad \text{(4) Correction factor}$$

[2] The cosines are calculated using the previously trained semantic space, in other words each of the terms to be compared is represented by a vector in this space. Any term vector might then be compared to another term vector using the cosine.

Where n is the total number of connections received by i (or the number of nodes in the neighbors layer) and C_i is the mean of all the strengths that node i received.

Finally, the 20 most highly activated neighbors are chosen from all of those activated in the Integration phase, and a new utterance is constructed with them. In other studies an utterance has been replaced by a more representative one [14], but now the aim is also for this new utterance to contain terms which are closer to the meaning originally intended by the user. These 20 terms are used to form a new pseudodocument, this time using Folding-in (see Figure 1), and they will be compared with the destination exemplars (created in the same way) in order to assign them a destination.

As the reader can realize, there are a few differences between the procedure to perform C-I used in this study and the original mechanism proposed by Kintsch [3]. Because call utterances are shorter and simpler than propositions within colloquial language, the algorithm used here is not exactly the original construction-integration algorithm. The integration part proposed by Kintsch is a spreading activation algorithm which is iterative until the net is stable (the cycle when the change in the mean activation is lower than a parameterized value), whereas our algorithm is a "one-shot" mechanism. The activation of each node is calculated based on the connections received. Another difference with the C-I algorithm as proposed by Kintsch is that we only consider words and not propositions nor situations. In any case, note that the original C-I is more complete and fine-grained, but our mechanism is sufficient for our purposes and may be more flexibly programmed, because an OOP (Object Oriented Programming) paradigm has been used, with classes such as net, layer, node, connection, etc., instead of the iterative vector × matrix multiplication in the original (see [39] for details of the original conception).

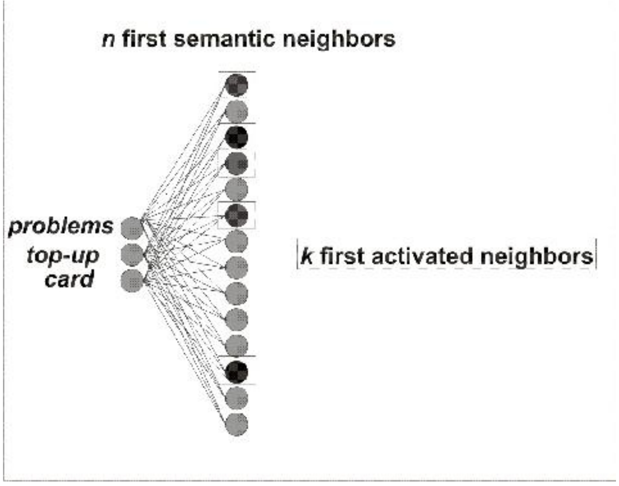

Figure 2: C-I based net implemented in this study. The K most strongly activated neighbors (with a square) will be represented in a pseudodocument.

5 Software and Architecture used

This section describes the application that we have implemented in our lab, and the auxiliary Software and technologies that has been used. In the last part, we describe the place of each module in the global architecture.

5.1 IVR Application

The IVR Application is implemented using VXML technology (Voice Extensible Markup Language), located on a Tomcat application server (icon D in Figure 3). The server dynamically generates VXML files and sends them to the VXML interpreter, in this case VoxPilot (icon B in Figure 3), as it requests them and according to the application flow. First, the interpreter requests the start and welcome VXML files, then the VXML where the user is asked to request a service ("say anything"). This second VXML has voice recognition and therefore requires a grammar - in this case a grxml grammar generated using the SLM. In this way, Voxpilot sends the user response to the voice recognition application (icon C in Figure 3) along with the route where the grammar is located on the application server. Finally, the ASR module returns the text recognized with its confidence interval. This recognized text is sent in the actual request to the next VXML located on the application server, using the SOAP protocol[3], to the semantic router (icon E in Figure 3). The semantic router returns a list with the four most likely destinations. The first destination from the list will be inserted into the VXML that is sent once again to Voxpilot for it to run the definitive Call Routing routine.

5.2 ASR Module

The recognition engine used is Nuance 9® (icon C in Figure 3), located on the Speech Server®, also by Nuance, with a test license that allows the use of 4 channels in non-production environments. This recognition engine accepts grammars programmed in the standard .grxml (SGRS Speech Grammar Recognition Specification) so it is a good match for our SLM.

5.3 Semantic Router

The router is basically dedicated to the task of receiving texts from the application server (icon D in Figure 3) and returning a list with the 4 most likely routes, or simply returning a rejection if it does not reach a confidence threshold. As we explained above, the router application is based in the LSA and C-I framework and was programmed in VB.NET using the object-oriented programming (OOP) paradigm. It is accessible as a web service on a Microsoft IIS server, which offers a series of functions and procedures such as loading the destination exemplars and converting them into vectors, and loading the reference semantic space (the semantic space

[3] **SOAP** (Simple Object Access Protocols) is a standard protocol that defines how two objects in different processes can communicate by exchanging XML data.

previously generated with the Gallito® tool mentioned above). It also has a configuration file which specifies the acceptance thresholds of the logistic function, the method of converting utterances into pseudodocuments (C-I Routing and its parameters or Direct Routing) and the classes that will be used according to the training (months, days, mobile phone brands, countries, etc.). The fact that it is implemented as a Web Service means it can be integrated into a SaaS (Software as a Service) structure that allows it to route service utterances in a centralized manner. When the destination has been assigned, this module returns a list with the four most likely destinations and their cosines.

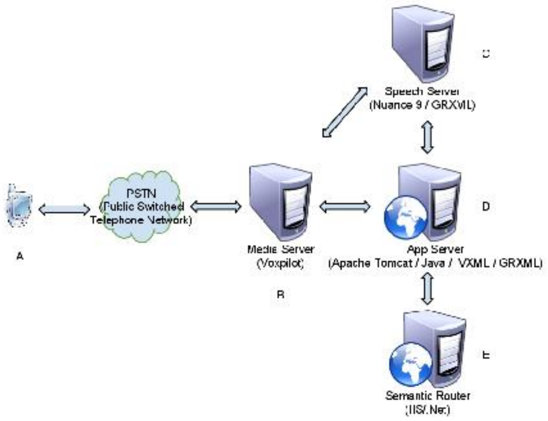

Figure 3: Software, technologies, modules and system architecture used in our lab.

6 Evaluation of the system

In this section we describe the procedure that we followed to examine the efficiency of the system in different configurations. First we will present the variables that were manipulated. Second, we will explain the procedure and the method to test them.

6.1 Dependent variable and independent variables

The dependent variable used to assess the efficiency of the system was whether the destination was correctly classified or not.

The first independent variable is called Routing Method and has to do with how LSA vectorially represents the test utterances and the utterances that represent the destinations. There are two methods (see section 4.2.3): Direct routing and C-I routing.

The second independent variable is the Number of Captures, the number of utterances captured by the voice recognition application that were passed as input to the categorization module. The voice recognition application can be programmed to return only one text containing what it has recognized of the participant's voice message. However, as this captured text derives from a probability-based model, it can also be programmed to return the second, or the third most probable text, and so on. Bearing in mind the idea that by combining the first two options we can create a text including more correctly recognized words than using only the first option, the independent variable contains two quantities: the first recognized utterance and the concatenation of the first and second utterance. We named the two levels Captur1 and Captur2.

A third independent variable is Accuracy, which is measured by a value F’ (see later in the method section). This value F’ is broken down into two groups: high and low accuracy (see later for a more detailed description).

Lastly, another independent variable is the number of tokens that is the length in words of transcriptions of the utterances. Two groups are again formed: Short and long utterances.

6.2 Procedure and Method

The evaluation method is as follows: 1,872 (randomized) audio files are obtained from real calls to several telecommunications companies. All of these audio files were then transcribed, to create the ideal condition where the ASR module captures the utterance perfectly. Recognition of each of these audio files was forced using the Nuance® ACC_TEST tool. The outputs from this tool provided the first two captures of what had been recognized (the two first outputs from the n-best list).

To obtain objective measures of the third independent variable, Accuracy, measured as the deviation between what is recognized and what is transcribed, Information Retrieval measures were used. Their usage is justified by some studies in substitution of the WER (Word Error Rate) [36]. The measures used for IR were Recall, Precision and the combination of both in F'. Recall is the proportion of the transcription that is present in recognition; Precision is the proportion of the recognition that is present in transcription. These two measures range from 0 to 1, and are combined in an index, F´. The latter is the harmonic mean of the precision and the recall measures. To calculate them, we used the most popular natural language package in the Python environment, the Natural Language Toolkit (NLTK) [37] available at http://www.nltk.org/. Using this tool we extracted recall, precision and F' of each recognition compared to its transcription as well as revealing the recognition loss, and these were introduced as variables in the overall analysis of the router (becoming the Accuracy variable). It is important to note that both precision and recall have been calculated using only relevant terms, meaning those that will be considered by the router. Function words and words from the stop-list, for example, will be excluded from the calculations. At the same time, we also counted the number of tokens in each transcription to obtain a measure of the fourth independent variable, which gave us an idea of the length of the phrases uttered by users to make system requests.

In the second phase the router is forced to categorize the text of each audio clip, beginning with the first recognition capture (Captur1), followed by the first two captures concatenated together (Captur2). This operation is repeated twice, once with the router working in Direct Routing mode, and once with C-I Routing. As a result we obtain all the combinations of independent variables (Routing Method × Number of Captures to route). In addition the router is forced to categorize the transcriptions, simply to have a baseline, but without introducing these results into the analysis matrix. The Call Routing is considered correct if the first destination returned by the router coincides with the ideal destination previously assigned to the utterance by a human[4].

With this data we now have all the necessary conditions and all the grouping variables. A matrix is formed and we proceeded to carry out a Repeated Measures ANOVA (Routing Method × Number of Captures to route) with two grouping variables (Accuracy – high or low, and Nº of tokens – long or short). In summary, a 2×2×2×2 ANOVA. The results will be extracted from this analysis, and their implications will be examined in the discussion.

At the end of the analysis, a logistic function is proposed to create acceptance zones in the router. There is a first zone where the first hypothesis returned and routed to this destination is accepted as valid; then a second disambiguation zone, where we might disambiguate between four hypotheses returned by the router, asking the user to choose;, and a third rejection zone , where any hypothesis is assumed to be erroneous and is not routed. The results of this mechanism will be shown as the percentage of correct call routings.

7 Results

In this section we will present the results of the study. First we will present the percentages of correctly routed calls under all possible conditions, including those which involve voice recognition. In addition, we also present the results of an ANOVA which displays the interactions between the variables involved: Routing Method, Accuracy, number of tokens and Number of Captures. Finally, the results of entering an acceptance criterion by means of a logistical function are presented, as well as the application of various confidence levels for a potential disambiguation strategy.

7.1 Performance of the ASR module (F')

As suggested above, one of the proposed means of measuring the ASR module's performance is applying Information Recall indices. To be precise, our system provides the following indices: Recall=.912, Precision=.959, F'=.932.

7.2 Decrease in performance caused by voice recognition

Another of the relevant issues to be examined is the lowered performance of the whole system when speech recognition output utterances are routed rather than transcriptions. If we consider only the router's first hypothesis (Table 1), the results show that between the best condition with transcriptions (.75 with C-I) and the best condition with recognition (.67 also with C-I), we lose 8 percentage points in the rate of correct choices. If we consider the first four hypotheses returned (Table 2), between the best condition with transcriptions, also using C-I (.91), and the best with recognition, again using C-I (.85), the difference is reduced to 6 percentage points. We should remember, though, that our application has not undergone any optimization process for either voice recognition or the model of categories and that the main aim of this study is to check the scenarios in which C-I might behave more productively. We will look at this in the following section.

	1st hypothesis		
	Trans	Captur1	Captur2
No C-I	.69	.63	.62
C-I	.75	.67	.66

Table 1: Percentages of correctly routed calls, considering only the first hypothesis returned by the router. Captur1 and Captur2 represent the conditions where one or two speech recognition hypotheses respectively are used.

	Accumulated first 4 hypotheses		
	Trans	Captur1	Captur2
No C-I	.89	.83	.83
C-I	.91	.85	.84

Table 2: Percentages of correctly routed calls, considering the four hypotheses returned by the router. Captur1 and Captur2 represent the conditions in which one or two speech recognition hypotheses respectively are used.

7.3 Results of the ANOVA

Another of the aims of this study was to analyze the performance of each method in different scenarios - specifically bearing in mind the Accuracy of recognition measured with F', and the number of tokens in user utterances. For this purpose the ANOVA described in the method section was carried out, extracting both main effects and interactions.

[4] Very often, many categories are interrelated or overlap, such that not even a human classifier could be sure which category to assign. As a last resort, whilst both may even be correct, the choice between one or another is binary and exclusive, so the results may be understated. Some evaluations of Call Routing have used not only binary coincidence to correct this but also ratings of the fitness of the destination returned, or also a scale of possible success [38]. We have not used this type of evaluation due to the high cost involved, which lessens information in the results and implies that they may be understated.

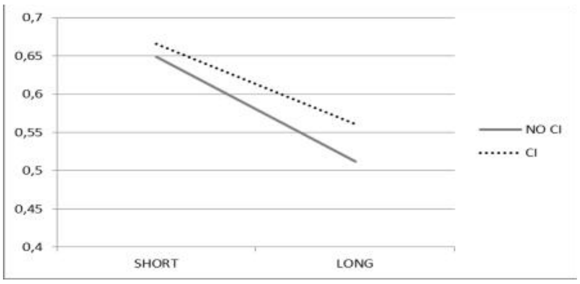

Figure 4: Interaction between Routing Method and Number of Tokens.

Three significant main effects were in fact found. The first of them relates to Routing Method (F(1,1735)=18.13, MSE=.087, p < .001). C-I is better than Direct Routing as a general effect. The second is the Nº of tokens (F(1,1735)=34.07, MSE=.637, p < .001). Short phrases boost the router's effectiveness. The third is F' (F(1,1735)=128.14, MSE=.637, p < .001). The Accuracy of voice recognition also increases the router's performance.

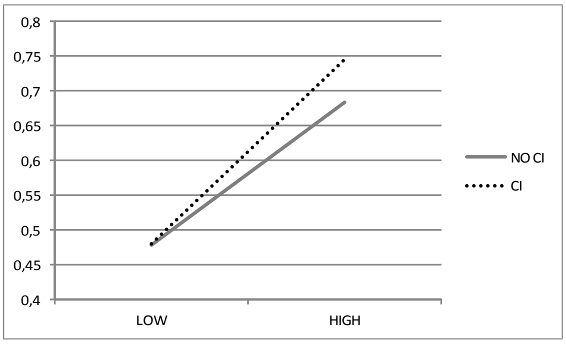

Figure 5: Interaction between Routing Method and Accuracy.

Given that we also found significant interaction effects with the variables above, we will focus on these, leaving the main effects as supplementary information. Firstly, we find a two-way interaction effect between the factors Routing Method and Nº of tokens (F(1.,1735)=4.45, MSE=.087, p = .035). The loss of effectiveness observed when the user utters long phrases is greater with Direct Routing than with C-I (Figure 4). Both methods show reduced effectiveness with long utterances, but C-I less so, hence we could say it has a corrective effect. We also found a significant effect for the two-way interaction between Routing Method and Accuracy (F(1,1735)=15.41, MSE=.087, p < .001). C-I shows beneficial effects if the quality of the recognition (measured by F') is good (Figure 5); otherwise, C-I works the same as Direct Routing. Lastly, we found a significant effect for the two-way interaction between Number of Captures and Accuracy (F(1,1735)=8.60, MSE=.087, p = .003). The results of this last interaction show how concatenating the first two speech recognition Captures is advantageous compared to using only one, if the quality of the recognition is low (Figure 6).

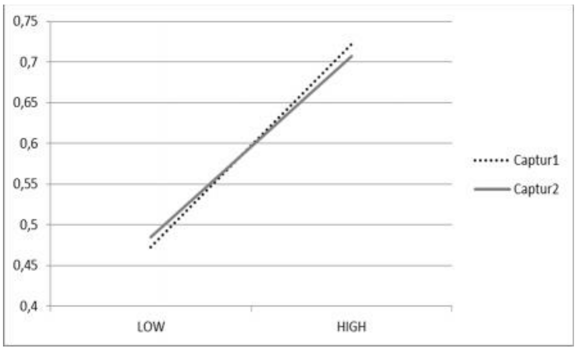

Figure 6: Interaction between Number of Captures and Accuracy.

7.4 Acceptance levels

Beyond straight performance data - in other words, the percentage of occasions a hypothesis returned is correct - we must introduce acceptance levels into the process. These maximize the correct choices and minimize errors by using a confidence threshold above which the route proposed by the system is accepted (formula 5). Previous studies have often introduced objective acceptance criteria into the process, such as logistic functions of acceptance [4]. In our case we will use the same parameters used by a similar study [1] in the logistic function, as they gave good results. In this function two parameters act as predictor variables, and the correct choice / error rate as our dependent variable. The parameters are as follows: firstly we use the cosine from the first hypothesis (average of the 4 largest cosines with the exemplars of this destination). As the second parameter, we use the difference between this first cosine and the cosine from the second hypothesis. The latter parameter takes into account the level of certainty about whether the first hypothesis should be returned rather than the second.

$$P(Y=1)=\frac{1}{1+e^{2.17-3.02d\cos-2.97\cos}}$$

(Formula 5)

With the output from the logistic function, we define three zones of acceptance. The first one (0.5 - 1), above which we accept the label directly; the second one (0.4 – 0.5), where we then disambiguate (using a question) between the four destinations proposed by the router (those that have the greatest cosine); and the third one (0 – 0.4), where it is directly rejected. In this way, not only does the percentage of correct choices fall within the acceptance zone, but it can also retrieve calls that are in the intermediate zone. Disambiguation mechanisms, such as preparing questions using the four hypotheses returned, could be built in this zone. For example, if the logistic function returned 0.46, the hypothesis returned would not be rejected, but rather disambiguated using the four hypotheses returned by the router, in the hope that the correct route would be among them, which is very likely. The results (Table 3) show that this disambiguation mechanism would be productive. In the acceptance zone (0.5 - 1), the percentage of correct destinations would be 76.54%. If we also disambiguated in the next zone of acceptance (0.4 – 0.5), we would guarantee that 81% of the time the correct destination would be among the four hypotheses used to ask the question. In addition, our data would contain 215 calls that would be rejected directly as they are in the rejection zone (0 - 0.4). This amounts to only 12% of the calls.

	Error		Correct choice	
	N	%	N	%
0.4 to 0.5 (disambiguation)	39	18.14	176	81.86
0.5 to 1 (1st hypothesis)	308	23.46	1005	76.54

Table 3: Correct choice rates in the two zones of acceptance.

8 Discussion

The overall results of this system are encouraging. Considering that it is a pilot study performed without optimizing voice recognition, and using a hypothetical customer service operation, the results are very satisfactory. To offer some raw data, 67% of utterances were assigned a destination where router and human agreed (with spontaneous speech recognition and categorization). As explained above, this finding is cautious if we consider the possibility that the destinations overlap or that there is ambiguity in the human assignment of an utterance to a destination. In addition, by introducing an acceptance criterion and using disambiguation mechanisms, a large proportion of the remaining utterances (81% of the total utterances) can be assigned to a correct destination. In any case, since this is an academic study, our aim was to focus on checking the performance of some methods in certain scenarios and not so much on the overall results, which could be improved upon.

Our system brings several features together. One of them is that the router is based on LSA, a technique that is independent from the subject domain and is also fairly economical in terms of implementation. We have inserted an additional layer in this router, based on the assumption that the meaning of an utterance is not the sum of its words. Rather it is important to take into account how these words activate others that are not explicit, but participate in the final meaning. Whilst it is not identical, this layer is based on the Construction-Integration model, which makes the same assumption. We have also decided to summarize the final routing destinations in a file of sample utterances that represent them (golden destinations), thus avoiding any change in the organization of destinations, leading to a need to reclassify all utterances in the training corpus. This provides added flexibility to the system and reduces the response times to changes. All this has been integrated with a voice recognition engine (supported by a Stochastic Language Model or SLM), whose outputs are passed to the router in order to assign destinations. In order to maximize the number of correctly recognized words, not only the first hypothesis output from the ASR module is routed: there is an option where the concatenated first and second hypotheses can also be routed. This was tested on audio clips of real calls and an experimental design that allowed us to evaluate performance in some scenarios depending on length of utterances and accuracy of voice recognition.

The analysis performed yielded various findings. Firstly that C-I is better than the direct method, in particular when it comes to cushioning the drop in performance in certain scenarios. It is true that in conditions where recognition has greatly declined, the contribution of C-I is not important, but when recognition is not bad, the C-I method seems to behave best with long utterances. Whilst this is not the case in a model like C-I, some previous studies have found benefits in long utterances if the speech recognition outputs are corrected by means of similarities extracted from LSA [24]. It should come as no surprise that in our system the C-I method performed best for long utterances, given that the original C-I model was created to account for longer propositions [2] and that the presence of a number of terms in the phrase facilitates the building of a context. This helps to over-weight the words that are within this context and to under-weight those that are not - for example, substitution or insertion errors. It also biases the meaning of ambiguous terms toward a meaning coherent with this context. The great contribution of this type of models is that they objectively mimic the process carried out in working memory while processing texts. As a text is being read or listened to, it is available in working memory, which retrieves content related with each word in the text from long-term memory. This will be the construction phase. In the integration phase, a mutual constraint mechanism is applied to this linked content [34] in order to extract the key idea. Therefore it is only to be expected that the higher the number of words in working memory (up to a threshold for simultaneous processing), the more data will be available to carry out this integration in a more correct way.

Secondly, although this is approximate complementary data, we have also tried using two voice recognition captures in the Call Routing process. The results are modest, although they show a subtle trend. When recognition is expected to decline, there is a tendency that taking two hypotheses rather than one improves the results. We sense that occasionally the errors committed in the first hypothesis are not committed in the second, and vice-versa.

Thirdly, we have seen that introducing a logistic function with some parameters helps to form acceptance criteria, above which correct choices are maximized, either by correctly rejecting the label or by accepting a label that later proves to be correct. By doing so, the results rise to 76.54% accuracy (either correct choices or correct rejections). We have also shown that improved performance results from setting up three zones of acceptance: the first one (0.5 - 1), above which we accept the label directly; the second one (0.4 – 0.5), where we then disambiguate (in the form of a question) between the four destinations proposed by the router; and the third one (0 - 0.4), where the label is rejected directly. In this way this, we achieve 79.3% correct Call Routing, and the correct destination of the utterances that remain in the intermediate zone for disambiguation would be among the four labels proposed in the question (the four returned by the router) in 81.56% of cases. Thus we have not only the percentage of correct choices in the acceptance zone, but also those that arise from disambiguation. It is clear that this requires a cost in terms of disambiguation question design, and a cost in terms of satisfaction, but it might be a good way to gradually implement the system.

There is one last thing to note: this system's adaptability to change. On the one hand, although in this study we have used labels to identify the destinations for utterances in the training corpus, acceptable results can be obtained without them. We could even extend the training corpus, combining labeled with unlabeled parts, or parts labeled using different criteria. If we did these things, we would find a small, controlled reduction in performance, but not an abrupt drop [1]. On the other hand, the only labels that need to be exhaustive are those that identify the utterances that act as destination exemplars. These exemplars form part of a chosen sample of training utterances, but represent only a small percentage of them. Since there are few of them, they can be examined, changed or expanded quickly. Thus any change in the routing model (for example, a re-dimensioning of skills) can be dealt with, with acceptable response times, and with no need to retrain all utterances, thereby slowing down the process.

9 Conclusion

In view of these results, the possibility of using psycholinguistic models in information recovery or utterance categorization systems is encouraging. Simulating human processing is not an easy task. It is not even easy to describe, but it is useful to reflect upon it in order to find possible improvements to current systems. In summary, LSA represents a very flexible and economical means of implementing Call Routing, and also allows us to explore algorithms and methods derived from research in Cognitive Science that may prove very promising. In this case, we have presented a system that combines LSA with a network based on Construction-Integration. The results obtained are good, although fine tuning is needed to optimize the voice recognition process, as well as more coherent organization of the destinations (which would be the case in a working production system). In any case, what has proved most interesting about this study is not the overall results in absolute terms, but rather testing the C-I layer and how it works. This has offered quite promising results, demonstrating superiority in some scenarios and stability in others. We believe that establishing links between computational science and psycholinguistics can help to find ways to optimize current categorization systems.

Acknowledgements

The authors wish to thank Airenas Vaiciunas from the Department of Applied Informatics at Vytautas Magnus University (Lithuania) for his useful comments about SLMs and also to Semantia Lab (www.semantialab.es) for supporting the logistic of this research.

References

[1] Jorge-Botana, G., Olmos, R. & Barroso, A. (2012). The Construction-Integration algorithm: A means to diminish bias in LSA-based Call Routing. *International Journal of Speech Technologies, 15(2)*: pp. 151-164.

[2] Kintsch, W. (1998). The use of knowledge in discourse processing: A construction integration model. *Psychological Review*, 95: 163-182

[3] Kintsch, W., & Welsch, D. *The construction-integration model: A framework for studying memory for text*. In W.E. Hockley & S. Lewandowsky (Eds.), Relating theory and data: Essays on human memory in honor of Bennet B. Murdock. Hillsdale, NJ: Erlbaum. 1991: 367-385.

[4] Chu-Carroll, J., y Carpenter, B. (1999). Vector-based natural language call routing, *Computational Linguistics, 25(3)*: pp. 361-388.

[5] Brumby, D. & Howes, A. Good enough but I´ll just check: Web page search as attentional refocusing. *Procedings of the 6th International Conference on Cognitive Modelling*, 46-51. 2004.

[6] Brumby,D. & Howes, A. Interdependence and past experience in menu choice assessment. In: Alterman, R., Kirsh, D. (Eds.), *Proceedings of the 25thAnnualConference of the Cognitive Society*. Lawrence Erlbaum Associates, Mahwah, NJ, p. 1320. 2003.

[7] Jurafsky, D. & and Martin, J. Speech and Language Processing: *An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics*. 2nd edition. Prentice-Hall. 2009.

[8] Gasanova, T., Zhukov E., Sergienko, R., Semenkin, E. & Minker, W. A Semi-supervised Approach for Natural Language Call Routing. *Proceedings of the 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*. August 2013. Metz, France.

[9] Sarikaya, R., Hinton, G. & Ramabhadran, B. Deep Belief Nets for Natural Language Call-Routing. *In ICASSP-2011*.

[10] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. Indexing. (1990). By Latent Semantic Analysis. *Journal of the American Society For Information Science, 41*: pp. 391-407.

[11] Landauer, T. K., & Dumais, S. T. A solution to Plato's problem: the Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. (1997). *Psychological Review, 104*, pp. 211-240.

[12] Jones, M.P., & Martin, J.H. Contextual spelling correction using latent semantic analysis. *In Proceedings of the Fifth Conference on Applied Natural Language Processing.* pp. 163-176. 1997.

[13] Bellegarda, J.R. Exploiting latent semantic information in statistical language modeling. *In Proceedings of the IEEE. 88 (8),* 1279– 1296. 2000

[14] Jen-Tzung Chien, Meng-Sung Wu and Hua-Jui Peng. (2004). Latent semantic language modeling and smoothing, *International Journal of Computational Linguistics and Chinese Language Processing, vol. 9*, no. 2, pp. 29-44.

[15] Pucher, M., Y. Huang, Y. Combination of latent semantic analysis based language models for meeting recognition. *In Proceedings of the Second IASTED International Conference on Computational Intelligence*, San Francisco, California, USA, November 20-22, 2006

[16] Lim, B.P, Ma, B., & Li, H. Using Semantic Context to Improve Voice Keyword Mining, *In Proceedings of the International Conference on Chinese Computing (ICCC 2005), 2005*; 21-23, March, Singapore

[17] Shi, Y. *An Investigation of Linguistic Information for Speech Recognition Error Detection*, Ph.D. University of Maryland, Baltimore County, Baltimore. 2008.

[18] Miller, G.A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM Vol. 38,* No. 11: 39-41.

[19] Fellbaum, C. (1998, ed.) *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.

[20] Wandmacher, T. & Antoine, J. Methods to Integrate a Language Model with Semantic Information for a Word Prediction Component. *In Proceedings of EMNLP-CoNLL*, 506-513. 2007.

[21] Cox, S. & Shahshahani, B. A Comparison of some Different Techniques for Vector Based Call-Routing. *In Proceedings of 7th European Conf. on Speech Communication and Technology*, Aalborg. 2001.

[22] Li, L. & Chou, W. Improving latent semantic indexing based classifier with information gain, *In Proceedings of the 7th International Conference on Spoken Language Processing, ICSLP-2002*, 1141-1144. September 16-20, 2002 Denver, Colorado, USA.

[23] Serafin, R. & Di Eugenio. B. (2004). FLSA: Extending Latent Semantic Analysis with features for dialogue act classification. *In Proceedings of ACL04, 42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain, July.

[24] Tyson, N. & Matula, V.C. Improved LSI-Based Natural Language Call Routing Using Speech Recognition Confidence Scores, *In Proceedings of EMNLP*. 2004.

[25] Stolcke, A. SRILM - An Extensible Language Modeling Toolkit, *In Proceedings of Intl. Conf. Spoken Language Processing*, Denver, Colorado, September 2002.

[26] Jorge-Botana, G., León, J.A., Olmos, R. & Hassan-Montero, Y. (2010). Visualizing polysemy using LSA and the predication algorithm. *Journal of the American Society for Information Science and Technology*, 61(8), pp. 1706–1724.

[27] Jorge-Botana, G., Olmos, R., León J.A. (2009) Using LSA and the predication algorithm to improve extraction of meanings from a diagnostic corpus. *Spanish Journal of Psychology*, 12(2), pp. 424-440.

[28] Jorge-Botana, G., León, J.A.,Olmos, R., & Escudero, I. (2010). Latent Semantic Analysis Parameters for Essay Evaluation using Small-Scale Corpora. *Journal of Quantitative Linguistics, 17(1),* pp. 1–29

[29] Suendermann, D., Hunter, P., Pieraccini, R., Call Classification with Hundreds of Classes and Hundred Thousands of Training Utterances ... and No Target Domain Data, in Perception in Multimodal Dialog Systems, *Proceedings of the 4th IEEE Tutorial and Research Workshop on Perception and Interactive Technologies for Speech-Based Systems*, PIT 2008, Kloster Irsee, Germany, June 16-18, 2008, Springer-Verlag, pp. 81-87.

[30] Kintsch, W., Patel, V., & Ericsson, K. A. (1999) The role of Long-term working memory in text comprehension. *Psychologia*, 42: 186-198.

[31] Kintsch, W. (2000) Metaphor comprehension: A computational theory. *Psychonomic Bulletin & Review, 7,* pp. 257-266.

[32] Kintsch, W. (2001). Predication. *Cognitive Science, 25,* pp. 173-202.

[33] Kintsch, W., & Bowles, A. (2002). Metaphor comprehension: What makes a metaphor difficult to understand? *Metaphor and Symbol, 17*, pp. 249-262.

[34] Kintsch, W. *Meaning in context*. In Landauer, T. K, McNamara, D., Dennis, S. & Kintsch, W. (Eds.) Handbook of Latent Semantic Analysis. Mahwah, NJ: Erlbaum. 2007: 89-105.

[35] Olmos, R., León, J.A., Jorge-Botana, G.,& Escudero I. (2009). New algorithms assessing short summaries in expository texts using Latent Semantic Analysis. *Behavior Research Methods*, 41, pp. 944-950.

[36] McCowan, Moore, Dines, Gatica-Perez, Flynn, Wellner & Bourlard. *On the use of information retrieval measures for speech recognition evaluation*. Technical Report. IAIDP. 2005.

[37] Bird, S. and Loper, E. NLTK: The Natural Language Toolkit. *In Proceedings of ACL04, 42nd Annual Meeting of the Association for Computational Linguistics*. Barcelona, Spain, July. 2004.

[38] Malmström P. E. *Methods for Evaluating a Natural Language Call Routing Application: A Case Study.* Master’s thesis of the Uppsala University. Department of Linguistics and Philology. 2010

An Exact Analytical Grossing-Up Algorithm for Tax-Benefit Models

Miroslav Verbič, Mitja Čok and Tomaž Turk
University of Ljubljana, Faculty of Economics,
Kardeljeva ploščad 17, 1000 Ljubljana, Slovenia

Keywords: deductive data imputation, household budget survey, microsimulation, tax-benefit models

Received: November 27, 2014

In this paper, we propose a grossing-up algorithm that allows for gross income calculation based on tax rules and observed variables in the sample. The algorithm is applicable in tax-benefit microsimulation models, which are mostly used by taxation policy makers to support government legislative processes. Typically, tax-benefit microsimulation models are based on datasets, where only the net income is known, though the data about gross income is needed to successfully simulate the impact of taxation policies on the economy. The algorithm that we propose allows for an exact reproduction of a missing variable by applying a set of taxation rules that are known to refer to the variable in question and to other variables in the dataset during the data generation process. Researchers and policy makers can adapt the proposed algorithm with respect to the rules and variables in their legislative environment, which allows for complete and exact restoration of the missing variable. The algorithm incorporates an estimation of partial analytical solutions and a trial-and-error approach to find the initial true value. Its validity was proven by a set of tax rule combinations at different levels of income that are used in contemporary tax systems. The algorithm is generally applicable, with some modifications, for data imputation on datasets derived from various tax systems around the world.

Povzetek: Članek predstavlja algoritem obrutenja, ki omogoča izračunavanje bruto dohodkov iz neto dohodkov ob širokem naboru davčnih pravil različnih davčnih sistemov. Algoritem omogoča reproduciranje manjkajočih spremenljivk in je široko uporaben pri mikrosimulacijskem modeliranju.

1 Introduction

There are various techniques for data imputation, which give to the researcher an opportunity to remedy the situation when the dataset is not complete. This does not come without costs, since data imputation can easily introduce biased parameter estimates in statistical applications [1, 2], or in other domains [3, 4]. Imputation techniques rely on deterministic and stochastic approaches, mostly under the assumption that the variable in question is in some way related to other variables under investigation. In this paper, we are exploring a case of deductive approach [5], a possibility to estimate a missing variable by applying a set of rules for which it is known that they refer to the variable in question and to other variables in the dataset. This set of rules might be enforced in various contexts, for instance by legislation, government policy, or other institutional or social constraints. If there is a consistent set of rules, which are enforced in practice, and the rules are comprehensive, the researcher could develop a formal algorithm with respect to the rules and variables in the dataset, which would allow for the data imputation of the missing variable.

Let us consider the case of household budget survey (hereinafter HBS) datasets. HBS surveys are implemented at the national level of EU member states [6], where taxpayers report their net income for different income sources (e.g. wages, rents, pensions), as well as socio-economic data, which enable estimations of tax allowances and tax credits. HBS datasets are most valuable in many microsimulation tax-benefit models. Such models are standard tools in academia, in financial industry, and for underpinning everyday policy decisions and government legislative processes [7, 8, 9, 10].

Gross income represents a starting point for any tax simulation (including tax-benefit modelling), but HBS datasets are usually reporting only net amounts. As noted in [7], one possibility to generate gross income is the statistical approach based on information on both net and gross income. Using this information, a statistical model can be developed that yields estimates of net/gross ratios. These estimates are then applied to net incomes in order to compute gross amounts.

The second known technique is the iterative algorithm that exploits the tax and contribution rules already built into tax-benefit models to convert gross income into net income [8, 9]. The procedure takes different levels of gross income for each observation, applies them to the tax rules, calculates the net income, and compares it with the actual net income as long as the gross income fits the actual net income within approximation limits.

Both techniques, namely the statistical approach and the iterative algorithm, give gross income values that are estimates and not the actual gross income values. The task is not trivial, since modern tax systems include various rules for taxation and their combinations, and they usually involve a bracketing system for one or more parameters. Involvement of bracketing systems (and especially their combination) means that the calculation of net income from gross income is analytically non-reversible function.

In this paper, we are presenting a solution to this problem, namely an algorithm that enables a full restoration of the gross income value. The algorithm includes a set of analytical inversions combined with a trial-and-error approach to deal with bracketing system combinations. The proposed algorithm allows for the calculation of gross income from net income for a broad set of taxation possibilities, where only information on net income is available, along with information on tax reliefs. The algorithm is feasible in cases of proportional and progressive tax schedules of personal income tax (hereinafter PIT) and social security contributions. It also covers tax allowances as well as tax credits. It is thereby generally applicable to contemporary tax systems around the world.

The validity and accuracy of the proposed algorithm was tested by its application to a synthetic sample of taxpayers using an artificial system of personal income tax (PIT) and social security contributions. A comparison of gross income, calculated from net income using the proposed technique, with the initial gross income demonstrates the complete accuracy of the algorithm.

The rest of the paper is organized as follows. In Section 2, we analyse taxation rules that are used in contemporary taxation systems. The analysis is a basis for the formalization of the imputation algorithm, which is explained in Section 3, including detailed solutions and proofs for various combinations of tax rules and bracketing systems. A test of the validity and accuracy of the proposed algorithm is presented in Section 4. In the Conclusion, the proposed algorithm is presented in its full form, which can be directly applied in practice.

2 Analysis of taxation rules

Gross income is a starting point for the taxation of personal income, as to which we can distinguish three basic approaches [11, 12]: comprehensive income tax, dual income tax, and flat tax. Under a comprehensive income tax system, all types of labour and capital incomes are taxed uniformly by the same progressive tax schedule. A dual income tax system retains progressive rates on labour income, while introducing a proportional tax rate on capital income, e.g. the Scandinavian dual income tax [13]. The third option, which has been dominating income tax reforms in Eastern Europe [14, 15], is the flat-tax concept, although it is noted that this concept has not been implemented in any country in Western Europe [6].

Hereby we follow the most comprehensive procedure for the taxation of gross income, which is presented in Table 1 and includes a combination of progressive tax schedules and flat rates, with the addition of tax allowances and tax credits.

Gross income
– Social security contributions[a] ▪ determined by social security contributions schedule ▪ set as a proportion of gross income ▪ given in absolute amounts – Other costs related to the acquisition of net income ▪ determined by a schedule ▪ set as a proportion of gross income ▪ given in absolute amounts – Tax allowances
= Personal income tax base × Personal income tax rate[b]
= Initial personal income tax – Personal income tax credit
= Final personal income tax
Net income = Gross income – Social security contributions – Final personal income tax

[a] Employee social security contributions

[b] Either a single (flat) tax rate or set by the tax schedule

Table 1: General procedure for taxing gross income.

Table 1 contains the general procedure for the taxation of gross income. From gross income, employee social security contributions and other costs related to the acquisition of income (e.g. travel allowances or standardized costs set as a proportion of gross income) are deducted. Further, the tax allowances are subtracted and the tax base is obtained, which is subject to a PIT calculation using the tax schedule or a proportional (flat) tax rate. In this way, the initial PIT is calculated, which could be further reduced by a tax credit in order to calculate the final PIT and the net income.

From the taxation point of view, other costs related to the acquisition of income have consequences identical to social security contributions or tax allowances. Therefore, our further development implicitly incorporates these costs into the concepts of social security contributions and tax allowances.

When the schedules are applied, the PIT schedule or social security contributions schedule consists of a number of tax brackets with different marginal tax rates. The amount of PIT is calculated from the tax base according to the PIT schedule. Likewise, the amount of social security contributions is determined by gross income and the social security contributions schedule.

In general, at the annual level tax bases from different income sources are summed up into a single tax base, which is subject to a single-rate schedule, and then the final annual PIT is calculated. An alternative option is a dual-tax system, where the PIT is calculated separately for different income sources (multiple-rate schedule).

The procedure from Table 1 covers the existing tax systems to a great extent. In several OECD countries [16], the employee social security contributions are determined by the schedule (i.e. Austria, France) or set as a proportion of gross income (i.e. Spain, Norway), while social security contributions set by absolute amount are not very common and can be found, e.g., in Slovenia for certain categories of the self-employed.

The algorithm applies the logic of social security contributions to the *Other costs related to the acquisition* *of net income* (i.e. cost connected with the real estate maintenance in the case of taxing income from rents) and tax allowances (i.e. for children or interest of housing loan allowances), which are found across the tax systems. The algorithm also covers the case of social security ceiling (i.e. in Austria, Germany). Regarding the calculation of PIT, the algorithm covers the prevailing progressive PIT schedule, as well as flat-tax systems (e.g. in Hungary or Bulgaria).

However, the algorithm (more precisely, equations that cover specific combinations of tax parameters) has to be adapted to certain country specifics, which are not explicitly set out by the procedure from Table 1. For example, if social security contributions are not included in the PIT base, the gross income shall be calculated by the algorithm assuming that social security contributions are zero. Another example refers to above mentioned social security contribution ceiling. In this case, a zero rate tax bracket of social security contribution schedule above the set ceiling should be applied in an appropriate equation that is suitable to the specific combination of tax parameters of the particular country.

The algorithm hereinafter is derived for the case when there is only one PIT instrument and one SSC instrument, thus two instruments in total. However, in actual fiscal systems there are cases when two or more PIT or SSC instruments are applied to a single income source at the same time. In these cases, the net to gross conversion is more complicated, since a "compression" of two (or more) PIT or SSC instruments should be done into a single PIT or SSC instrument.

During the year, when a particular income source is paid out, the advance (in-year) PIT is usually paid at the time of disbursing the income source. This advance PIT is taken into account once the final annual PIT is calculated (i.e. the advance PIT is consolidated with the annual PIT). This procedure is called withholding.

Understanding the mechanism of (in-year) advance PIT is important when we are dealing with survey data, such as HBS datasets. In a typical survey, respondents report their net income from different income sources for a certain period of the year, when their income sources are only subject to (in-year) advance PIT. In order to calculate the overall annual gross income, the reported net income from different income sources should be initially grossed-up using the algorithms that take account of various rules of advance (in-year) PIT and social security contributions for each income source separately. The focus of our paper is the development of these grossing-up algorithms for different income sources. Once the grossed-up amounts from different income sources are calculated, they can be summed up into overall taxpayers' annual gross income, which is the starting point for building a microsimulation model.

Thus, the calculation of gross income from net income of a single income source (i.e. calculation of gross wages from given net wages) thus depends on different combinations of tax parameters from Table 1, which are described in detail as an algorithm in the paper. For example, in a case when gross wages are subject of: (a) progressive PIT schedule, (b) social security contributions, which are set as a proportion of gross wage with a ceiling, (c) tax allowances, and (d) without tax credits (e.g. in-year taxation of wages in Croatia), then equation (9) should be applied. Since the ceiling of social security contributions is set, this implies that the applied value of social security contribution rate above the ceiling should be zero.

Table 1 can be transformed into the following expression:

$$N = G - S - PIT\,, \tag{1}$$

where N and G represent net and gross income, respectively, S is the sum of social security contributions, and PIT is the personal income tax.

Social security contributions S are a function of gross income. Similarly, PIT is a function of the tax base, which is the difference between gross income (reduced by social security contributions) and tax allowances, TA. This can be generalized as follows:

$$N = G - f_S(G) - f_{PIT}(G - f_S(G) - TA)\,. \tag{2}$$

Function $f_S(G)$ can be defined in practice in different ways. A common approach is to use a schedule system, but it can also be defined as a proportion of gross income or as an absolute amount.

In practice, function $f_{PIT}(G - f_S(G) - TA)$ is usually defined by a schedule system (different from the schedule system for social security contributions). As mentioned, function f_{PIT} can also incorporate the concept of a tax credit.

Our task is to estimate gross income G from expression (2) from the known values of N and TA and from a set of constraints that are usually given by social security contributions and PIT schedule systems, or by other legislative rules. The combination of two schedule systems makes solving equation (2) for G particularly challenging. The solution we propose in this paper has a trial-and-error nature. The idea is to prepare a set of all possible (PIT and social security contribution) bracket combinations. Then, we calculate for each taxpayer 'candidate' gross income values for each bracket combination, calculate net incomes from these candidate gross incomes, and compare the results to the starting value of net income. The gross income candidate that fits (or equals) the net income is the true gross income value. The fit is exact, i.e. we find the actual gross income in a non-iterative way.

The following section describes the construction and design of the procedures we propose to deal with different income sources taxed by different rules. The general setup of the grossing-up algorithm is explained. Sections 3.2 and 3.3 set out a detailed examination of various taxation rules for social security contributions and tax crediting, together with the proposed grossing-up procedures for specific tax rule combinations.

3 Data imputation algorithm

In this section, we explore a general setup where the tax system involves a combination of the following elements: (1) a social security contributions schedule, (2) a PIT schedule, and (3) tax allowances. This general setup forms the basis for development of the proposed algorithm. In the next steps, we incorporate other tax complexities, i.e. other rules for calculating social security contributions and various rules for determining tax credits.

Function $f_S(G)$ can be expanded by the rules of the social security contributions schedule to:

$$f_S(G) = S = Sr_s(G - L_s) + \sum_{j=1}^{s-1} Sr_j(H_j - L_j)\,, \qquad (3)$$

i.e. for each bracket, social security contributions are equal to the social security contributions marginal rate Sr_s, multiplied by the difference between gross income G and the lower bracket margin (s denotes the social security contributions bracket). This amount is added to the social security contributions, which are collected for all 'lower' brackets (i.e. brackets from 1 to s – 1). H_s and L_s denote the upper and lower social security bracket margins.

Similarly, function f_{PIT} can be expanded by the rules of the schedule system for PIT to:

$$\begin{aligned} f_{PIT}(G) = PIT = Tr_b\left(G - f_S(G) - TA - L_b\right) + \\ + \sum_{i=1}^{b-1} Tr_i\left(H_i - L_i\right), \end{aligned} \qquad (4)$$

where Tr_b is the marginal tax rate for PIT bracket b, L_b is the lower margin for bracket b, Tr_i is the marginal rate for bracket i, and H_i and L_i are the upper and lower margins of bracket i, respectively.

By combining (3) and (4), we obtain:

$$\begin{aligned} N_{sb} &= G_{sb} - \left(Sr_s(G_{sb} - L_s) + \sum_{j=1}^{s-1} Sr_j(H_j - L_j) \right) - \\ &- \left(Tr_b \left(G_{sb} - \left(Sr_s(G_{sb} - L_s) + \sum_{j=1}^{s-1} Sr_j(H_j - L_j) \right) - \right.\right. \\ &\left.\left. - TA - L_b \right) + \sum_{i=1}^{b-1} Tr_i\left(H_i - L_i\right) \right), \end{aligned} \qquad (5)$$

The above equation holds for an individual taxpayer, when PIT was calculated by the tax authorities in such a way that social security bracket s corresponds to gross income G, and PIT bracket b corresponds to (G – S – TA). Since we do not know the actual G and S, we cannot directly establish, which PIT and social security contributions brackets (and corresponding marginal rates) were actually used for each individual taxpayer by the tax authorities.

By reordering expression (5), we can express gross income as follows:

$$\begin{aligned} G_{sb} = \frac{N_{sb} - Sr_s L_s - Tr_b L_t + Sr_b Tr_b L_s}{\left(Sr_s - 1\right)\left(Tr_b - 1\right)} - \\ - \frac{Tr_b TA + Tr_b \Sigma_s - \Sigma_s - \Sigma_b}{\left(Sr_s - 1\right)\left(Tr_b - 1\right)}, \end{aligned} \qquad (6)$$

where

$$\Sigma_b = \sum_{i=1}^{b-1} Tr_i\left(H_i - L_i\right)$$

and

$$\Sigma_s = \sum_{j=1}^{s-1} Sr_s\left(H_s - L_s\right).$$

Following our general trial-and-error scheme, the grossing-up algorithm is as follows:

1. For each statistical unit, calculate the matrix with $K \cdot B$ candidate gross incomes as its elements, according to equation (6):

$$\left\{ \begin{matrix} G_{11} & \cdots & G_{K1} \\ & \ddots & \\ \vdots & G_{kl} & \vdots \\ & & \ddots \\ G_{1B} & \cdots & G_{KB} \end{matrix} \right\},$$

   where K and B are the number of social security contributions brackets and the number of PIT brackets, both defined by the PIT and social security contributions system, respectively, and where $k = 1, \ldots, K$ and $l = 1, \ldots, B$.

2. Calculate the net incomes from the matrix of candidate gross incomes according to the tax rules:

$$\left\{ \begin{matrix} N_{11} & \cdots & N_{K1} \\ & \ddots & \\ \vdots & N_{kl} & \vdots \\ & & \ddots \\ N_{1B} & \cdots & N_{KB} \end{matrix} \right\}.$$

3. In the above matrix, find the net income N_{kl}, which is equal to the starting net income for this individual taxpayer: $N = N_{kl}$.

4. The actual gross income G for this individual taxpayer is then: $G = G_{kl}$.

In the next subsections, we discuss the following extensions to this general setup: (1) social security contributions are not determined by the schedule system, but as a proportion of gross income or as an absolute amount (Section 3.1), and (2) tax credits are included according to various rules for their determination (Section 3.2).

3.1 Variations of social security contributions

In the following section, we extend the general setup to include cases where social security contributions are not determined by a schedule, but as a proportion of gross income or as an absolute amount.

3.1.1 Social security contributions as a proportion of gross income

When social security contributions are set as a proportion of gross income, equation (2) can be rewritten as:

$$N = \left(1 - Sr\right)G - f\left(\left(1 - Sr\right)G - TA\right), \tag{7}$$

where Sr is the rate of social security contributions, expressed as a proportion of the gross income. By simplifying equation (5), we obtain:

$$\begin{aligned} N_b = {} & \left(1 - Sr\right)G_b - \Big(Tr_b\left(\left(1 - Sr\right)G_b - TA - L_b\right) + \\ & + \sum_{i=1}^{b-1} Tr_i\left(H_i - L_i\right)\Big), \end{aligned} \tag{8}$$

which holds for each PIT bracket b. By reordering, we can express the gross income with the equation:

$$G_b = \frac{N_b - Tr_b TA - Tr_b L_b + \Sigma_b}{\left(1 - Sr\right) - Tr_b\left(1 - Sr\right)}. \tag{9}$$

From here, we can proceed according to the general setup, outlined above.

3.1.2 Social security contributions as an absolute amount

When social security contributions are set as an absolute amount, we can simplify equation (5):

$$\begin{aligned} N_b = {} & G_b - S - \Big(Tr_b\left(G_b - S - TA - L_b\right) + \\ & + \sum_{i=1}^{b-1} Tr_i\left(H_i - L_i\right)\Big) \end{aligned} \tag{10}$$

and by reordering we obtain:

$$G_b = \frac{N_b + S - Tr_b S - Tr_b TA - Tr_b L_b + \Sigma}{1 - Tr_b}. \tag{11}$$

From here, we can proceed according to the general setup, outlined above.

3.2 Grossing-up procedure when PIT is subject to a tax credit

A tax credit means that PIT is reduced by a certain amount (called a tax credit) and that the gross income source is effectively not taxed with the full PIT (the 'initial PIT'), but with the PIT reduced by the amount of the tax credit (the 'final PIT'). In practice, if a tax credit is calculated to be greater than the initial PIT, then net income N equals gross income G, as the net income cannot exceed the gross income (i.e. a tax credit can be as high as the initial PIT).

In various tax systems, a tax credit can be defined in three ways: (1) as a proportion of the initial PIT, (2) as a proportion of the gross income, or (3) as an absolute amount.

3.2.1 Tax credit as a proportion of the initial PIT

In general, we can express a tax credit as a proportion of the initial PIT as:

$$\begin{aligned} N = {} & G - f_S(G) - f_{PIT}(G - f_S(G) - TA) + \\ & + c_{PIT} \cdot f_{PIT}(G - f_S(G) - TA), \end{aligned} \tag{12}$$

where c_{PIT} is the share of the tax credit in the initial PIT. Following the above, we can write:

$$\begin{aligned} N_{sb} = {} & G_{sb} - \left(Sr_s(G_{sb} - L_s) + \sum_{j=1}^{s-1} Sr_j(H_j - L_j)\right) - \\ & - \Bigg(Tr_b\Bigg(G_{sb} - \left(Sr_s(G_{sb} - L_s) + \sum_{j=1}^{s-1} Sr_j(H_j - L_j)\right) - \\ & - TA - L_b\Bigg) + \sum_{i=1}^{b-1} Tr_i\left(H_i - L_i\right)\Bigg) + \\ & + c_{PIT}\Bigg(Tr_b\Bigg(G_{sb} - \left(Sr_s(G_{sb} - L_s) + \sum_{j=1}^{s-1} Sr_j(H_j - L_j)\right) - \\ & - TA - L_b\Bigg) + \sum_{i=1}^{b-1} Tr_i\left(H_i - L_i\right)\Bigg), \end{aligned} \tag{13}$$

which holds for a specific combination of social security contributions and PIT brackets. Solving (13) for G, we obtain:

$$\begin{aligned} G_{sb} = {} & \frac{N_{sb} - Sr_b L_s - Tr_b L_t + cTr_b L_t + Tr_b Sr_b L_s}{\left(1 - Sr_s\right)\left(Tr_b\left(c_{PIT} - 1\right) + 1\right)} - \\ & - \frac{c_{PIT} Tr_b Sr_b L_s + Tr_b TA - c_{PIT} Tr_b TA}{\left(1 - Sr_s\right)\left(Tr_b\left(c_{PIT} - 1\right) + 1\right)} - \\ & - \frac{\Sigma_s\left(Tr_b\left(1 - c_{PIT}\right) - 1\right) + (c_{PIT} - 1)\Sigma_b}{\left(1 - Sr_s\right)\left(Tr_b\left(c_{PIT} - 1\right) + 1\right)}. \end{aligned} \tag{14}$$

When the tax credit is set as a proportion of the initial PIT and social security contributions are defined by a schedule, the above equation should be used instead of expression (6) in the general setup.

Tax credit as a proportion of the initial PIT and social security contributions as a proportion of the gross income

Where social security contributions are set as a proportion of the gross income, the general procedure can be simplified. In this case, the net income can be expressed as:

$$N=(1-Sr)G-f\left((1-Sr)G-TA\right)+ \\ +c_{PIT}\cdot f\left((1-Sr)G-TA\right). \tag{15}$$

The following equation holds for a particular tax bracket *b*:

$$N_b=(1-Sr)G_b-\left(Tr_b\left((1-Sr)G_b-TA-L_b\right)+\right. \\ \left.+\sum_{i=1}^{b-1}Tr_i\left(H_i-L_i\right)\right)+c_{PIT}\left(Tr_b\left((1-Sr)G_b-\right.\right. \\ \left.\left.-TA-L_b\right)+\sum_{i=1}^{b-1}Tr_i\left(H_i-L_i\right)\right). \tag{16}$$

By reordering we obtain:

$$G_b=\frac{N_b-\left(1-c_{PIT}\right)\left(Tr_bL_b+Tr_bTA-\Sigma_b\right)}{(1-Sr)(1-Tr_b+c_{PIT}Tr_b)}. \tag{17}$$

Thus, when a tax credit is set as a proportion of the initial PIT and social security contributions are set as a proportion of the gross income the above equation should be used instead of expression (6) in the general setup.

Tax credit as a proportion of the initial PIT and social security contributions as an absolute amount

If social security contributions are set as an absolute amount, we can redefine equation (10) to incorporate tax credit as a proportion of the initial PIT:

$$N_b=G_{sb}-S-\left(Tr_b\left(G_b-S-TA-L_b\right)+\right. \\ \left.+\sum_{i=1}^{b-1}Tr_i\left(H_i-L_i\right)\right)+c_{PIT}\left(Tr_b\left(G_b-S-TA-L_b\right)+\right. \\ \left.+\sum_{i=1}^{b-1}Tr_i\left(H_i-L_i\right)\right) \tag{18}$$

and by reordering we obtain:

$$G_b=\frac{N_b+S+\left(c_{PIT}-1\right)\left(Tr_bL_b+Tr_bS+Tr_bTA-\Sigma_b\right)}{\left(1-Tr_b+c_{PIT}Tr_b\right)}. \tag{19}$$

Thus, when a tax credit is set as a proportion of the initial PIT and social security contributions are set in an absolute amount, the above equation should be used instead of expression (6) in the general setup.

3.2.2 Tax credit as a proportion of the gross income

If the amount of a tax credit is defined as a proportion of the gross income, the net income calculation can be formalized as:

$$N=G-f_S(G)-f_{PIT}(G-f_S(G)-TA)+c_G\cdot G\,, \tag{20}$$

where c_G is the tax credit share of the gross income. For clarity, we can denote the initial PIT as:

$$PIT_I=f_{PIT}(G-f_S(G)-TA) \tag{21}$$

and the final PIT as:

$$PIT_F=f_{PIT}(G-f_S(G)-TA)-c_G\cdot G\,. \tag{22}$$

Since a tax credit can be as high as the initial PIT, the following rule applies:

$$N=\begin{cases}G-f_S(G)-PIT_F & \text{if} \quad c_G\cdot G<PIT_I;\\ G-f_S(G) & \text{if} \quad c_G\cdot G\geq PIT_I.\end{cases} \tag{23}$$

Due to this rule, the gross income cannot be easily estimated from net income *N* and tax allowances *TA*, as PIT_I and $c_G\cdot G$ are not known at this stage. The rule implies that the actual calculation of net income *N* for each taxpayer was done by the tax authorities either by:

$$N=G-f_S(G)-f_{PIT}(G-f_S(G)-TA)+c_G\cdot G \tag{24}$$

when $c_G\cdot G<f_{PIT}(G-f_S(G)-TA)$, or by:

$$N=G-f_S(G) \tag{25}$$

when $c_G\cdot G\geq f_{PIT}(G-f_S(G)-TA)$.

When we are interested in *G*, we can use these two approaches in reverse fashion (calculating *G* and not *N*), but we do not know which one, (24) or (25), is correct.

Let us consider the case when we calculate *G* for a particular taxpayer from known values of *N*, *TA* and the PIT schedule (as in Table 1), once by using the rule expressed in equation (24) and once by using the rule expressed in (25). We obtain two estimates for the taxpayer's gross income *G*:

$$G'=N+f_S(G)+f_{PIT}(G-f_S(G)-TA)-c_G\cdot G \tag{26}$$

and

$$G''=N+f_S(G)\,. \tag{27}$$

If the net income *N* for this particular taxpayer was actually calculated according to expression (24), this inequality holds true:

$$\begin{aligned}&\left(N+f_S(G)+f_{PIT}(G-f_S(G)-TA)-\right.\\&\left.\quad -c_G\cdot G\right)>\left(N+f_S(G)\right),\end{aligned} \tag{28}$$

since $c_G\cdot G<f_{PIT}(G-f_S(G)-TA)$ must hold. By using (26) and (27), we obtain:

$$G'>G''\,. \tag{29}$$

The proper value of gross income *G* is G', since net income *N* for this particular taxpayer was actually calculated according to expression (24).

Let us consider the opposite case where net income *N* for our taxpayer was actually calculated (by the tax authorities) according to (25). In this case, we can write:

$$\begin{aligned}&\left(N+f_S(G)+f_{PIT}(G-f_S(G)-TA)-\right.\\&\left.\quad -c_G\cdot G\right)\le\left(N+f_S(G)\right),\end{aligned} \tag{30}$$

since $c_G\cdot G\ge f_{PIT}(G-f_S(G)-TA)$ must hold. By using (26) and (27), we obtain:

$$G'\le G''\,. \tag{31}$$

The proper value of gross income *G* in this case is G''. Following (29) and (31), we can conclude that in both cases the highest value of G' and G'' is the one that actually holds:

$$G=\max\left(G',G''\right) \tag{32}$$

or

$$\begin{aligned}G=\max\Big(&\left(N+f_S(G)+f_{PIT}(G-f_S(G)-TA)-\right.\\&\left.-c_G\cdot G\right),\left(N+f_S(G)\right)\Big).\end{aligned} \tag{33}$$

For the construction of a general setup in the case of tax credits given as a proportion of the gross income, where social security contributions and PIT are calculated according to their schedules, we need to express equation (33) in a more exact way, for a specific combination of social security contribution and PIT brackets. The specific form for equation (26) is then:

$$\begin{aligned}N_{sb}=G_{sb}&-\left(Sr_s(G_{sb}-L_s)+\sum_{j=1}^{s-1}Sr_j(H_j-L_j)\right)-\\&-\left(Tr_b\left(G_{sb}-\left(Sr_s(G_{sb}-L_s)+\sum_{j=1}^{s-1}Sr_j(H_j-L_j)\right)-\right.\right.\\&\left.\left.-TA-L_b\right)+\sum_{i=1}^{b-1}Tr_i\left(H_i-L_i\right)\right)-c_G G_{sb}\end{aligned} \tag{34}$$

and for equation (27):

$$N_{sb}=G_{sb}-Sr_s(G_{sb}-L_s)+\sum_{j=1}^{s-1}Sr_j(H_j-L_j)\,. \tag{35}$$

From expression (34) we obtain:

$$\begin{aligned}G'_{sb}=&\frac{N_{sb}-Sr_sL_s-Tr_b\Sigma_s-Tr_bL_b}{1+c_G-Sr_b-Tr_b+Sr_bTr_b}+\\&+\frac{Sr_bTr_bL_s-Tr_bTA+\Sigma_s+\Sigma_b}{1+c_G-Sr_b-Tr_b+Sr_bTr_b}\end{aligned} \tag{36}$$

and from expression (35):

$$G''_{sb}=\frac{N_{sb}-Sr_bL_s+\Sigma_s}{1-Sr_b}\,. \tag{37}$$

According to expression (32), we can establish the right value for gross income G_{sb}:

$$G_{sb}=\max\left(G'_{sb},G''_{sb}\right). \tag{38}$$

Tax credit as a proportion of the gross income and social security contributions as a proportion of the gross income

Where social security contributions are set as a proportion of the gross income, the calculation of net income *N* for each taxpayer was done by the tax authorities either by:

$$N=(1-Sr)G-f_{PIT}((1-Sr)G-TA)+c_G\cdot G \tag{39}$$

when $c_G\cdot G<f_{PIT}((1-Sr)G-TA)$, or by:

$$N=(1-Sr)G \tag{40}$$

when $c_G\cdot G\ge f_{PIT}((1-Sr)G-TA)$. The reasoning is similar to that above where we constructed equations (34) and (35). These two equations can be simplified since we only have one social security contributions rate *Sr*, and we obtain:

$$\begin{aligned}N_b=\left(1-Sr\right)G_b&-\Big(Tr_b\left(\left(1-Sr\right)G_b-TA-L_b\right)+\\&+\sum_{i=1}^{b-1}Tr_i\left(H_i-L_i\right)\Big)+c_GG_b\end{aligned} \tag{41}$$

and

$$N_b=\left(1-Sr\right)G_b\,. \tag{42}$$

From this, we obtain two solutions for G_b:

$$G'_b=\frac{N_b-Tr_bL_b-Tr_bTA+\Sigma_b}{1+c_G-Sr-Tr_b+SrTr_b} \tag{43}$$

and

$$G_b'' = \frac{N_b}{1 - Sr}, \tag{44}$$

which should be used in the general setup instead of (36) and (37), respectively. Again, the matrix of candidate solutions is one-dimensional (a vector for gross income candidates, i.e. one value for each PIT bracket), since there is only one social security contributions rate.

Tax credit as a proportion of the gross income and social security contributions as an absolute amount

In this case, the procedure can follow the same principles we used to construct equations (34) and (35). Since social security contributions are now set as an absolute amount, these two equations can be simplified:

$$\begin{aligned} N_b = G_b - S - \Big(& Tr_b\left(G_b - S - TA - L_b\right) + \\ & + \sum_{i=1}^{b-1} Tr_i\left(H_i - L_i\right)\Big) + c_G G_b \end{aligned} \tag{45}$$

and

$$N_b = G_b - S\,. \tag{46}$$

The gross income for both cases can then be calculated from:

$$G_b' = \frac{N_b + S - Tr_b L_b - Tr_b S - Tr_b TA + \Sigma_b}{1 + c_G - Tr_b} \tag{47}$$

and

$$G_b'' = N_b + S\,, \tag{48}$$

which should be used in the general setup instead of (36) and (37), respectively.

3.2.3 Tax credit as an absolute amount

If the amount of a tax credit is defined as an absolute amount, the procedure is similar to the one described in Section 2.3.2. The net income can be expressed as:

$$N = G - f_S(G) - f_{PIT}(G - f_S(G) - TA) + C\,, \tag{49}$$

where C is the amount of the tax credit. The initial PIT is the same as in Section 2.3.2, equation (21), and the final PIT is:

$$PIT_F = f_{PIT}(G - f_S(G) - TA) - C\,. \tag{50}$$

The following rule applies:

$$N = \begin{cases} G - f_S(G) - PIT_F & \text{if} \quad C < PIT_I; \\ G - f_S(G) & \text{if} \quad C \geq PIT_I. \end{cases} \tag{51}$$

If net income N for a particular taxpayer was actually calculated (by the tax authorities) according to $C < PIT_I$ in (51), this inequality holds true:

$$\begin{aligned} & \left(N + f_S(G) + f_{PIT}(G - f_S(G) - TA) - C\right) > \\ & \qquad > \left(N + f_S(G)\right), \end{aligned} \tag{52}$$

since $C < f_{PIT}(G - f_S(G) - TA)$ must hold. In the opposite case, i.e. if net income N was calculated according to $C \geq PIT_I$, then:

$$\begin{aligned} & \left(N + f_S(G) + f_{PIT}(G - f_S(G) - TA) - C\right) \leq \\ & \qquad \leq \left(N + f_S(G)\right), \end{aligned} \tag{53}$$

since $C \geq f_{PIT}(G - f_S(G) - TA)$ must hold.

Following a similar reasoning to that in Section 2.3.2, we can conclude that the actual gross income G for a particular taxpayer must be:

$$\begin{aligned} G = \max\Big(& \big(N + f_S(G) + f_{PIT}(G - f_S(G) - \\ & - TA) - C\big), \left(N + f_S(G)\right)\Big) \end{aligned} \tag{54}$$

Quantity G' can be estimated from:

$$\begin{aligned} N_{sb} = G_{sb} & - \Bigg(Sr_s(G_{sb} - L_s) + \sum_{j=1}^{s-1} Sr_j(H_j - L_j) \Bigg) - \\ & - \Bigg(Tr_b \Bigg(G_{sb} - \Bigg(Sr_s(G_{sb} - L_s) + \sum_{j=1}^{s-1} Sr_j(H_j - L_j) \Bigg) - \\ & - TA - L_b \Bigg) + \sum_{i=1}^{b-1} Tr_i\left(H_i - L_i\right) \Bigg) + C \end{aligned} \tag{55}$$

and solving for G_{sb}:

$$\begin{aligned} G_{sb}' = & \frac{N_{sb} + C + Sr_s L_s + Tr_b L_b - Sr_s Tr_b L_s}{\left(Sr_s - 1\right)\left(Tr_b - 1\right)} + \\ & + \frac{Tr_b TA + Tr_b \Sigma_s - \Sigma_s - \Sigma_b}{\left(Sr_s - 1\right)\left(Tr_b - 1\right)}, \end{aligned} \tag{56}$$

whereas the estimation of G'' is already explained in (36) and (38).

We can conclude that in cases where the amount of a tax credit is defined as an absolute amount, the general setup is the same as that described in Section 2.3.2, except for equation (36), which should be substituted by equation (56).

Tax credit as an absolute amount and social security contributions as a proportion of the gross income

Where when social security contributions are set as a proportion of the gross income, the calculation of net income N for each taxpayer was done by the tax authorities either by:

$$N = (1 - Sr)G - f_{PIT}((1 - Sr)G - TA) + C \tag{57}$$

when $C < f_{PIT}((1 - Sr)G - TA)$, or by:

$$N = (1 - Sr)G \tag{58}$$

when $C \geq f_{PIT}((1 - Sr)G - TA)$. The reasoning is similar to that in Section 2.3.2. Equation (41) can be rewritten in the following form:

$$\begin{aligned} N_b = {} & (1 - Sr)G_b - \Big(Tr_b\left((1 - Sr)G_b - TA - L_b\right) + \\ & + \sum_{i=1}^{b-1} Tr_i\left(H_i - L_i\right)\Big) + C \end{aligned} \tag{59}$$

and from this, we can obtain:

$$G_b' = \frac{N_b - C - Tr_b L_b - Tr_b TA + \Sigma_b}{(Sr - 1)(Tr_b - 1)}, \tag{60}$$

which should be used in the general setup instead of (43), whereas equation (44) also applies in this case for obtaining G_b''. Again, the matrix of candidate solutions is one-dimensional (a vector for gross income candidates, i.e. one value for each PIT bracket).

Tax credit as an absolute amount and social security contributions as an absolute amount

In this case, the procedure can follow the same principle we introduced in Section 2.3.2. Since social security contributions are given as an absolute amount, equation (34) can be written in this way:

$$\begin{aligned} N_b = {} & G_b - S - \Big(Tr_b\left(G_b - S - TA - L_b\right) + \\ & + \sum_{i=1}^{b-1} Tr_i\left(H_i - L_i\right)\Big) + C, \end{aligned} \tag{61}$$

whereas equation (46) also holds in the case social security contributions are set as an absolute amount. The gross income can then be calculated from (61) as:

$$G_b' = \frac{N_b - C + S - Tr_b L_b - Tr_b S - Tr_b TA + \Sigma_b}{1 - Tr_b}, \tag{62}$$

which should be used in the general setup instead of (36), together with (48), which was derived from (46).

3.3 Algorithm in its full form

For clarity, the grossing-up procedure that we developed in the above sub-sections is given below, including all combinations of the taxation rules that we described in the subsections following the basic setup at the beginning of Section 3.

1. For each statistical unit, calculate the matrix of $K \cdot B$ candidate gross incomes according to equation (6):

$$G = \begin{Bmatrix} G_{11} & \cdots & G_{K1} \\ \vdots & G_{kl} & \vdots \\ G_{1B} & \cdots & G_{KB} \end{Bmatrix},$$

   where K and B are the number of social security contribution brackets and the number of tax brackets, both defined by the tax and social security contribution systems, respectively, where $k = 1, \ldots, K$ and $l = 1, \ldots, B$.

   Formulas for specific combinations of taxation rules can be found in Table 2.

   In cases where only the tax schedule system is used and social security contributions related to the acquisition of the income are set as one parameter, the above matrix of candidate gross incomes becomes a vector $\{G_1, \ldots, G_l, \ldots, G_B\}$.

2. Calculate the net incomes from the matrix of candidate gross incomes according to the tax rules:

$$N = \begin{Bmatrix} N_{11} & \cdots & N_{K1} \\ \vdots & N_{kl} & \vdots \\ N_{1B} & \cdots & N_{KB} \end{Bmatrix}$$

   or

$$\{N_1, \ldots, N_l, \ldots, N_B\}.$$

3. In the above matrix, find net income N_{kl} (or N_l), which is equal to the starting net income:

$$N = N_{kl} \text{ (or } N = N_l\text{)}.$$

4. The actual gross income G is then:

$$G = G_{kl} \text{ (or } G = G_l\text{)}.$$

System without tax credits			Equation
I	Schedule for social security contributions	$G_{sb}=\frac{N_{sb}-Sr_sL_s-Tr_bL_t+Sr_bTr_bL_s-Tr_bTA-Tr_b\Sigma_s+\Sigma_s+\Sigma_b}{\left(Sr_s-1\right)\left(Tr_b-1\right)}$	(6)
II	Social security contributions as a proportion of gross income	$G_b=\frac{N_b-Tr_bTA-Tr_bL_b+\Sigma_b}{\left(1-Sr\right)-Tr_b\left(1-Sr\right)}$	(9)
III	Social security contributions as an absolute amount	$G_b=\frac{N_b+S-Tr_bS-Tr_bTA-Tr_bL_b+\Sigma}{1-Tr_b}$	(11)
Tax credit as a proportion of the initial PIT			Equation
IV	Schedule for social security contributions	$G_{sb}=\frac{N_{sb}-Sr_bL_s-Tr_bL_t+c_{PIT}Tr_bL_t+Tr_bSr_bL_s-c_{PIT}Tr_bSr_bL_s-Tr_bTA}{\left(1-Sr_s\right)\left(Tr_b\left(c_{PIT}-1\right)+1\right)}+$ $+\frac{c_{PIT}Tr_bTA-\Sigma_s\left(Tr_b\left(1-c_{PIT}\right)-1\right)-(c_{PIT}-1)\Sigma_b}{\left(1-Sr_s\right)\left(Tr_b\left(c_{PIT}-1\right)+1\right)}$	(14)
V	Social security contributions as a proportion of gross income	$G_b=\frac{N_b-\left(1-c_{PIT}\right)\left(Tr_bL_b+Tr_bTA-\Sigma_b\right)}{(1-Sr)(1-Tr_b+c_{PIT}Tr_b)}$	(17)
VI	Social security contributions as an absolute amount	$G_b=\frac{N_b+S+\left(c_{PIT}-1\right)\left(Tr_bL_b+Tr_bS+Tr_bTA-\Sigma_b\right)}{\left(1-Tr_b+c_{PIT}Tr_b\right)}$	(19)
Tax credit as a proportion of gross income			Equation
VII	Schedule for social security contributions	$G_{sb}=\max\left(G'_{sb},G''_{sb}\right)$ $G'_{sb}=\frac{N_{sb}-Sr_sL_s-Tr_b\Sigma_s-Tr_bL_b+Sr_bTr_bL_s-Tr_bTA+\Sigma_s+\Sigma_b}{1+c_G-Sr_b-Tr_b+Sr_bTr_b}$ $G''_{sb}=\frac{N_{sb}-Sr_bL_s+\Sigma_s}{1-Sr_b}$	(38) (36) (37)
VIII	Social security contributions as a proportion of gross income	$G_b=\max\left(G'_b,G''_b\right)$ $G'_b=\frac{N_b-Tr_bL_b-Tr_bTA+\Sigma_b}{1+c_G-Sr-Tr_b+SrTr_b}$ $G''_b=\frac{N_b}{1-Sr}$	(38) (43) (44)
IX	Social security contributions as an absolute amount	$G_b=\max\left(G'_b,G''_b\right)$ $G'_b=\frac{N_b+S+Tr_bL_b-Tr_bS-Tr_bTA+\Sigma_b}{1-c_G-Tr_b}$ $G''_b=N_b+S$	(38) (47) (48)
Tax credit as an absolute amount			Equation
X	Schedule for social security contributions	$G_{sb}=\max\left(G'_{sb},G''_{sb}\right)$ $G'_{sb}=\frac{N_{sb}+C+Sr_sL_s+Tr_bL_b-Sr_sTr_bL_s+Tr_bTA+Tr_b\Sigma_s-\Sigma_s-\Sigma_b}{\left(Sr_s-1\right)\left(Tr_b-1\right)}$ $G''_{sb}=\frac{N_{sb}-Sr_bL_s+\Sigma_s}{1-Sr_b}$	(38) (56) (37)
XI	Social security contributions as a proportion of gross income	$G_b=\max\left(G'_b,G''_b\right)$ $G'_b=\frac{N_b-C-Tr_bL_b-Tr_bTA+\Sigma_b}{\left(Sr-1\right)\left(Tr_b-1\right)}$ $G''_b=\frac{N_b}{1-Sr}$	(38) (60) (44)
XII	Social security contributions as an absolute amount	$G_b=\max\left(G'_b,G''_b\right)$ $G'_b=\frac{N_b-C+S-Tr_bL_b-Tr_bS-Tr\,_bTA+\Sigma_b}{1-Tr_b}$ $G''_b=N_b+S$	(38) (62) (48)

Table 2: Equations for specific combinations of taxation rules.

4 Results and discussion

Table 3 presents a summary of all possible social security contributions and tax credit combinations explored in Section 3. In reality, for any income source one of these combinations is applicable. Parallel to this, the PIT schedule system and tax allowances in absolute amounts are assumed.

Our approach can also be applied to flat PIT systems (i.e. with a single proportional PIT rate). If this is a case, we apply only one PIT bracket with a positive marginal PIT rate. Where tax allowances are not set as absolute amounts, they can be expressed as an 'additional layer' of social security contributions.

To test for the validity and accuracy of the proposed algorithm, we created a synthetic sample of 10,000 taxpayers with a normally distributed gross income, where the mean gross income was 50,000 mu (monetary units) and the standard deviation was 11,500 mu. We assumed the following tax parameters:

1. The PIT schedule includes three brackets:
   - 0 – 20,000 mu, a 15% marginal PIT rate;
   - 20,000 – 50,000 mu, a 25% marginal PIT rate;
   - over 50,000 mu, a 45% marginal PIT rate.
2. The social security schedule includes three brackets:
   - 0 – 10,000 mu, a 17% marginal rate;
   - 10,000 – 40,000 mu, a 20% marginal rate;
   - over 40,000 mu, a 0% marginal rate.
3. Social security contributions as a proportion of the gross income were set at 22%.
4. Social security contributions as an absolute amount were set at 500 mu.
5. Tax allowances were set at an absolute amount of 2,000 mu.
6. The amounts of tax credits were given as follows: 13% of the gross income, 6% of the initial PIT, or 200 mu.

These parameters were applied to the entire population of taxpayers according to the general procedure for taxing gross income (Table 1) and the specific combination of tax rules from Table 3.

In the first step, we generated the amount of gross income for each taxpayer. In the second step, we calculated the net income according to combinations I to XII (from Table 3) of the tax rules, as is done in practice by tax authorities.

In the third step, we applied the proposed grossing-up algorithm to combinations I–XII for each taxpayer. Finally, we compared the grossed-up income with the initial gross income.

	Schedule for social security contributions	Social security contributions as a proportion of the gross income	Social security contributions as an absolute amount
System without tax credits	I	II	III
Tax credit as a proportion of the initial PIT	IV	V	VI
Tax credit as a proportion of the gross income	VII	VIII	IX
Tax credit as an absolute amount	X	XI	XII

Table 3: Summary of tax rules combinations (detailed equations are given in Table 4).

The comparison of the gross income, calculated from the net income using the grossing-up algorithm, with the initial gross income demonstrates the complete accuracy of the algorithm for all income types.

As an example, we can repeat the steps for an individual taxpayer with a gross income equal to 49,433.10 mu. In the second step, we calculated the net income amount for all 12 combinations of the tax rules (see Table 4).

	Schedule for social security contributions	Social security contributions as a proportion of the gross income	Social security contributions as an absolute amount
System without tax credits	33,799.80 (I)	31,418.30 (II)	39,199.80 (III)
Tax credit as a proportion of the initial PIT	34,275.80 (IV)	31,846.70 (V)	39,783.80 (VI)
Tax credit as a proportion of the gross income	40,226.10 (VII)	37,844.60 (VIII)	45,626.10 (IX)
Tax credit as an absolute amount	33,999.80 (X)	31,618.30 (XI)	39,399.80 (XII)

Table 4: Net income for a chosen taxpayer with G = 49,433.10 mu.

For each net income from Table 4, we applied the grossing-up algorithm (i.e. equations from Table 2). According to the technique, several gross income candidates were calculated for each of these net incomes.

Due to space limitations, here we (arbitrarily) present the gross income candidates for net income VII:

$$G^{VII} = \begin{Bmatrix} 48{,}465.20 & 50{,}134.40 & 48{,}465.20 \\ 49{,}907.60 & 51{,}371.40 & 49{,}907.60 \\ 47{,}926.10 & 49{,}433.10 & 47{,}926.10 \end{Bmatrix}.$$

To each of these gross income candidates we applied the taxation rules (in this case the combination of taxation rules VII) and calculated the net income:

$$N^{VII} = \begin{Bmatrix} 39{,}374.40 & 40{,}843.20 & 39{,}374.40 \\ 40{,}643.70 & 41{,}931.80 & 40{,}643.70 \\ 38{,}900.00 & 40{,}226.10 & 38{,}900.00 \end{Bmatrix}.$$

By comparing the elements of matrix N^{VII} with the net income for a combination of tax rules VII from Table 4, which equals 40,226.10 (VII), we identified the matching element in the third row and the second column. The corresponding gross income in matrix G^{VII} equals 49,433.10, which is identical to the initial gross income of this particular taxpayer. In other words, for this combination of tax rules (VII), the proposed grossing-up algorithm is accurate.

We repeated such tests for all 12 tax rule combinations and for 10,000 individual cases.

5 Conclusion

In this paper, we presented a detailed construction of deterministic data imputation algorithm. In particular, we described an exact grossing-up algorithm for calculating the pre-tax income from data, which are only available in net (after-tax) form, and proved its successfulness, since it leads to a complete data reconstruction.

Contemporary tax systems are rich in complexity, and some of tax rules combinations might not be covered by our technique. However, we believe that the general architecture of our proposition is sound and flexible enough to incorporate (with some modifications) additional, locally specific tax rules.

In general, if a set of rules that relate to the variables under investigation could be assembled, researchers and policy makers can perform data imputation in deterministic fashion, and construct the algorithm for the exact analytical generation of the missing values.

In future research efforts, a framework for feasibility assessment of such approach could be envisioned, which would employ estimates on rules' consistency and complexity on the one hand, and measures of the quality of replicated data on the other hand.

References

[1] Rancourt, E. (2007). Assessing and dealing with the impact of imputation through variance estimation. *Statistical Data Editing: Impact on Data Quality*. New York: United Nations.

[2] Rueda, M. M., Gonzalez, S. & Arcos, A. (2005). Indirect methods of imputation of missing data based on available units. *Applied Mathematics and Computation* 164: 249–261.

[3] Smirlis, Y. G., Maragos, E. K. & Despotis, D. K. (2006). Data envelopment analysis with missing values: An interval DEA approach. *Applied Mathematics and Computation* 177: 1–10.

[4] Raghunathan, T. E., Lepkowski, J. M., van Hoewyk, J. & Solenberger, P. (2001). A Multivariate Technique for Multiply Imputing Missing Values Using a Sequence of Regression Models. *Survey Methodology* 27: 85–95.

[5] Franklin, S. & Walker, C. (2003). *Survey methods and practices*. Ottawa: Statistics Canada.

[6] Fuest, C., Peichl, A. & Schaefer, T. (2008). Is a flat tax reform feasible in a grown-up democracy of Western Europe? A simulation study for Germany. *International Tax and Public Finance* 15: 620–636.

[7] Immervoll, H. & O'Donoghue, C. (2001). Imputation of gross amounts from net incomes in household surveys: an application using EUROMOD, *EUROMOD Working Papers* EM1/01. Colchester: ISER-Institute for Social and Economic Research.

[8] D'Amuri, F. & Fiorio, C. V. (2009). Grossing-Up and Validation Issues in an Italian Tax-Benefit Microsimulation Model. *Econpubblica Working Paper,* 117, Milano: University of Milan.

[9] Betti, G., Donatiello, G. & Verma, V. (2011). The Siena microsimulation model (SM2) for net-gross conversion of EU-silc income variables. *International Journal of Microsimulation* 4: 35–53.

[10] ISER – Institute for Social and Economic Research, https://www.iser.essex.ac.uk/euromod (April 16th, 2012)

[11] OECD – Organisation for Economic Co-operation and Development (2006). Reforming Personal Income Tax. *Policy Brief March*. Paris: OECD.

[12] Zee, H. H. (2005). Personal income tax reform: Concepts, issues, and comparative country developments. *IMF Working Paper* 87. Washington: International Monetary Fund.

[13] Sorenson, P. B. (2005). Dual income tax: Why and how? *FinanzArchiv* 61: 559–586.

[14] Ivanova, A., Keen, M. & Klemm, A. (2005). The Russian 'flat tax' reform. *Economic Policy* 20: 397–444.

[15] Moore, D. (2005). Slovakia's 2004 tax and welfare reforms. *IMF Working Paper* 133, Washington: International Monetary Fund.

[16] OECD – Organisation for Economic Co-operation and Development (2013). *Taxing Wages 2011–2012*. Paris: OECD.

The slWaC Corpus of the Slovene Web

Tomaž Erjavec
Dept. of Knowledge Technologies, Jožef Stefan Institute
Jamova cesta 39, Ljubljana
E-mail: tomaz.erjavec@ijs.si

Nikola Ljubešić
Faculty of Humanities and Social Sciences, University of Zagreb
Ivana Lučiča 3, Zagreb, Croatia
E-mail: nikola.ljubesic@ffzg.hr

Nataša Logar
Faculty of Social Sciences, University of Ljubljana
Kardeljeva ploščad 5, Ljubljana
E-mail: natasa.logar@fdv.uni-lj.si

Keywords: language technologies, corpus linguistics, World Wide Web, Slovene language

Received: February 25, 2015

The availability of large collections of text (language corpora) is crucial for empirically supported linguistic investigations of various languages; however, such corpora are complicated and expensive to collect. In recent years corpora made from texts on the World Wide Web have become an attractive alternative to traditional corpora, as they can be made automatically, contain varied text types of contemporary language, and are quite large. The paper describes version 2 of slWaC, a Web corpus of Slovene containing 1.2 billion tokens. The corpus extends the first version of slWaC with new materials and updates the corpus compilation pipeline. The paper describes the process of corpus compilation with a focus on near-duplicate removal, presents the linguistic annotation, format and accessibility of the corpus via Web concordancers. It then investigates the content of the corpus using the method of frequency profiling, by comparing its lemma and part-of-speech annotations with three corpora: the first version of slWaC, with Gigafida, the one billion word reference corpus of Slovene, and KRES, the hundred million word reference balanced corpus of Slovene.

Povzetek: Dostopnost velikih zbirk besedil (jezikovnih korpusov) je nujna za empirično podprte jezikoslovne raziskave posameznih jezikov, vendar pa je izdelava takih korpusov draga in zamudna. Korpusi besedil, zajetih s spleta, so v zadnjem času postali popularen vir jezikovnih vsebin, saj jih lahko zgradimo avtomatsko, vsebujejo pester nabor sodobnih besedilnih zvrsti in so zelo veliki. Prispevek predstavlja drugo različico korpusa slWaC, spletnega korpusa slovenščine, ki vsebuje 1,2 milijardi pojavnic. Korpus dopolnjuje prvo različico slWaC z novimi besedili, pridobljenimi z izboljšanimi orodji za zajem. V prispevku opišemo izdelavo korpusa s poudarkom na odstranjevanju podobnih vsebin ter jezikoslovno označevanje, format korpusa in njegovo dostopnost prek konkordančnika. Nato raziščemo vsebino korpusa z uporabo metode frekvenčnega profila, pri katerem leme in oblikoskladenjske oznake druge različice korpusa slWaC primerjamo s tremi korpusi: s prvo različice korpusa slWaC, z referenčnim korpusom Gigafida, ki vsebuje milijardo besed, in s stomilijonskim referenčnim uravnoteženim korpusom KRES.

1 Introduction

Large collections of digitally stored and uniformly encoded texts – language corpora – have for a number of years been the basic data resources that linguists, including lexicographers, have used for their investigations into language and for making dictionaries. However, the traditional way of compiling corpora, which involved acquiring texts from authors and publishers, which exists in many disparate formats, was very expensive in terms of time and labour.

With the advent of the Web, a vast new source of linguistic information has emerged. The exploitation of this resource has especially gained momentum with the WaCky initiative [1], which has popularised the concept of “Web as Corpus”. It has also made available tools for compiling such corpora and produced large WaC corpora for a number of major European languages. Now such corpora are also being built for the so called smaller languages, such as

Norwegian [8], Czech [18] and Serbian [11], moving the concept of a "large corpus" for smaller languages up to the 1 billion token frontier.

As Web corpus acquisition is much less controlled than that for traditional corpora, the necessity of analysing their content gains in significance. The linguistic quality of the content is mostly explored through word lists and collocates [1] while the content itself is explored using unsupervised methods, such as clustering and topic modelling [17].

For Slovene, a Web corpus has already been built [12]. However, the first version of slWaC (hereafter slWaC$_1$) was rather small, as it contained only 380 million words. Furthermore, it contained domains from the Slovene top-level domain only, i.e. only URLs ending with ".si" were harvested. In the meantime, hrWaC, the Croatian Web corpus had already moved to version 2, touching the 2 billion token mark, and Web corpora for Serbian and Bosnian were built as well [11], all of them passing the size of slWaC$_1$, making it high time to move forward also with slWaC.

This paper presents version 2 of slWaC (hereafter slWaC$_2$) which tries to overcome the limitations of slWaC$_1$: it extends it with a new crawl, which also includes well known Slovene Web domains from other top-level domains, and introduces a new pipeline for corpus collection and cleaning, resulting in a corpus of 1.2 billion tokens with removed near-duplicate documents and flagged near-duplicate paragraphs.

The rest of the paper is structured as follows: Section 2 presents the corpus construction pipeline, Section 3 introduces the linguistic annotation of the corpus, its format and its availability for on-line concordancing, Section 4 investigates the content of the corpus, by comparing it to slWaC$_1$, to the balanced corpus of Slovene KRES, and the reference corpus of Slovene Gigafida, while Section 5 gives some conclusions and directions for future work.

2 Corpus construction

2.1 Crawling

For performing the new crawl we used the SpiderLing crawler[1] with its associated tools for guessing the character encoding of a Web page, its content extraction (boilerplate removal), language identification and near-duplicate removal [19].

The SpiderLing crawler uses the notion of *yield rate* to optimize the crawling process regarding the amount of unique textual material retrieved given the overall amount of data retrieved. Yield rate is calculated for each Web domain as the ratio of bytes of text contributed to the final corpus and the bytes retrieved from that domain. Web domains with a yield rate under a predefined threshold are discarded from further crawling, thereby focusing the remaining crawl on the domains where more unique textual material is to be found. SpiderLing has two predefined yield rates that control when a low-yield-rate Web domain is blacklisted; we used the lower one which is recommended for smaller languages.

As seed URLs we used the home pages of Web domains obtained during the construction of slWaC$_1$ and additionally 30 well known Slovene Web domains, which are outside the .si top-level domain.

The crawl was run for 21 days, with 8 cores used for document processing, which includes guessing the text encoding, text extraction, language identification and physical duplicate removal, i.e. removing copies of identical pages which appear under different URLs. After the first 14 days there was a significant decrease in computational load, showing that most of the domains had been already harvested and that the process of exhaustively collecting textual data from the extended Slovene top-level domain was almost finished.

After completing the crawling process, which already included document preprocessing, we merged the new crawl with slWaC$_1$. We added the old dataset to the end of the new one, thereby giving priority to new data in the following process of near-duplicate removal. It should be noted that the corpus can, in cases when the content has changed, contain two texts with the same URL but with different crawl dates.

2.2 Near-duplicate removal

We performed near-duplicate identification both on the document and the paragraph level using the onion tool[2] with its default settings, i.e. by calculating 5-gram overlap and using the 0.5 duplicate content threshold. We removed the document-level near-duplicates entirely from the corpus, while keeping paragraph-level near-duplicates, labelling them with a binary attribute on the `<p>` element. This means that the corpus still contains the (near)duplicate paragraphs, which is advantageous for showing contiguous text from Web pages, but if, say, language modelling for statistical machine translation were to be performed [10], near-duplicate paragraphs can easily be removed.

The resulting size of the corpus (in millions of tokens) after each of the three duplicate removal stages is given in Table 1. We compare those numbers to the ones obtained on the Croatian, Bosnian and Serbian domains [11], showing that the second versions of the corpora (hrWaC and slWaC), which merge two crawls obtained with different tools and were collected three years apart, show a smaller level of reduction (around 30%) at each step of near-duplicate removal, while the first versions of corpora (bsWaC and srWaC), obtained with SpiderLing only and in one crawl, suffer more data loss in this process (around 35-40%).

[1] http://nlp.fi.muni.cz/trac/spiderling

[2] https://code.google.com/p/onion/

	PHY	DND	PND	R1	R2
slWaC$_2$	1,806	**1,258**	**895**	0.31	0.29
hrWaC$_2$	2,686	1,910	1,340	0.29	0.30
bsWaC$_1$	722	429	288	0.41	0.33
srWaC$_1$	1,554	894	557	0.42	0.37

Table 1: Sizes of the Web corpora in millions of tokens after removing physical duplicates (PHY), document near-duplicates (DND) and paragraph near-duplicates (PND), with the reduction ratio (R1 and R2) after the DND and subsequent PND steps.

2.3 Linguistic annotation

slWaC$_2$ was tagged and lemmatised with ToTaLe [4] trained on JOS corpus data [5]. However, it should be noted that ToTaLe had been slightly updated, so in particular the tokenisation of slWaC$_1$ and slWaC$_2$ at times differs. The morphosyntactic descriptions (MSDs) that the words of the corpus are annotated with follow the JOS MSD specifications, however, these do not define a tag for punctuation. As practical experience has shown this to be a problem, we have introduced a punctuation category and MSD, named “Z” in English and “U” in Slovene.

3 Overview of the corpus

3.1 Size of the corpus

Table 2 gives the size of slWaC$_2$, showing separately the amount of information from the 2011 crawl, from the 2014 crawl, and overall amount of information. For each of the counted elements we give the size of the corpus after removing document near-duplicates (DND from Table 1), and for the corpus which has also paragraph near-duplicates removed (PND).

Starting with the number of domains, it can be seen that the new crawl produced less domains than the first one, due to a large number (of the complete space of URLs) of static domains being removed in the physical deduplication stage (PHY). Nevertheless, the complete corpus has, in comparison to slWaC$_1$, about 12,000 new domains. Observing the URLs, we note that the new crawl gave somewhat less URLs than the old one, and that there is little overlap between the two, i.e. about 1%: 28,315 URLs are the same from both crawls, which means that their content has changed in the last three years (and are then in the corpus distinguished by having a different crawl date).

Regarding the number of paragraphs, we give both the numbers for DND and PND, with the reduction being very similar to the reduction on the token level already expressed in Table 1, i.e. 29%. For paragraphs, sentences, words and tokens, the complete corpus is simply the sum of the items for each of the two crawls. The most important numbers are the sizes of the complete corpus in tokens, i.e. 1.25 billion words for the DND and 900 million for PND, which makes the corpus almost as large as Gigafida [13], the largest corpus of Slovene to date.

3.2 Corpus format

The annotated corpus is stored in the so called vertical format, used by many concordancing engines. This is an XML-like format in that it has opening and closing or empty (structural) XML tags, but the tokens themselves are written one per line, with the first (tab separated) column giving the token (word or punctuation) itself, the second (in our case) its lemma (or, for punctuation, again the token), the third its MSD in English and the fourth the MSD in Slovene, as illustrated by Figure 1.

```
<text domain="www.cupradan.si"
      url="http://www.cupradan.si/"
      crawled="2014">
<gap extent="1000+"/>
<p type="text" duplicate="0">
<s>
*       *       Z         U
Izmed   izmed   Sg        Dr
vseh    ves     Pg-mpg    Zc-mmr
<g/>
,       ,       Z         U
ki      ki      Cs        Vd
boste   biti    Va-f2p-n  Gp-pdm-n
delili  deliti  Vmpp-pm   Ggnd-mm
video   video   Ncmsan    Sometn
...
```

Figure 1: Vertical format of the annotated slWaC$_2$.

The example also shows a few other features of the encoding. Each text is given its URL, the domain of this URL and the year (2011 or 2014) on which it was crawled. Boilerplate removal often deletes linguistically uninteresting texts from the start (and end) of the document, which is marked by the empty gap element, which also gives the approximate extent of the text removed. The paragraphs are marked by their type, which can be “heading” or “text”, while the “duplicate” attribute tells whether the paragraph is a (near) duplicate of some other paragraph in the corpus, in which case its value is “1”, and “0” otherwise. Finally, we also have the empty “glue” element g, which can be used to suppress the space between two adjacent tokens in displaying the corpus.

3.3 Availability

The corpus is mounted under the noSketchEngine concordancer [15] installed at nl.ijs.si/noske. The concordancer allows for complex searches in the corpus, from concordances taking into account various filters, to frequency lexica over regular expressions.

slWaC$_2$	2011	2014	All
Domains	25,536	22,062	37,759
URLs (DND)	1,528,352	1,295,349	2,795,386
Paragraphs (DND)	7,535,453	18,303,123	25,838,576
(PND)	6,325,075	10,329,692	16,654,767
Sentences (DND)	22,615,610	50,693,747	73,309,357
(PND)	19,001,653	31,560,289	50,561,942
Words (DND)	360,273,022	718,332,186	1,078,605,208
(PND)	301,547,669	465,780,456	767,328,125
Tokens (DND)	421,178,853	837,727,874	1,258,906,727
(PND)	352,474,874	542,912,192	895,387,066

Table 2: Size of the slWaC 2.0 corpus.

We also make the corpus available for download, but not directly, mainly due to question of personal data protection. Namely, the corpus contains most of the Slovene Web, at least in the .si domain, so it also contains a lot of personal names with accompanying text. This is not such a problem with the concordancer, as similiar results on Web-accessible personal names can be also obtained by searching through Google or the Slovene search engine Najdi.si. However, being able to analyse the complete downloaded corpus enables much more powerful information extraction methods to be used, potentially leading to abuse of personal data. This is why we make the corpus available for research only, and require a short explanation of the use it will be put to. However, we make available the metadata of the corpus, in particular the list of URLs included in it, which enables other to make their own corpus on this basis.

4 Comparative corpus analysis

This section investigates how different the slWaC$_2$ corpus is from its predecessor, slWaC$_1$ and from two other corpora of Slovene [13]: the balanced reference corpus KRES, which contains 100 million words, and the reference corpus Gigafida, which contains 1.2 billion words, mostly (77%) from printed periodicals created between 1990 and 2011. The KRES corpus was sampled from Gigafida and has roughly the following structure: 35% books, 40% periodicals and 20% Internet. To establish how different these corpora are we used the method of frequency profiling [14]. We first made a frequency lexicon of the annotation under investigation (lemma or grammatical description) for slWaC$_2$ and the corpus it was compared with, and then for each item in this lexicon computed its log-likelihood (LL). The formula takes into account the two frequencies of the element as well as the sizes of the two corpora which are being compared; the greater LL is, the more the item is specific for one of the corpora. To illustrate, we give in Table 3 the first 15 lemmas with their LL score and their frequency per million words in slWaC$_1$ and slWaC$_2$, with the larger frequency in bold.

Lemma	LL	slWaC$_1$pm	slWaC$_2$pm
član	30,366	0.131	**0.282**
foto	23,092	0.018	**0.081**
m2	22,826	0	**0.033**
biti	22,767	**76,984**	74,493
°	21,447	0.001	**0.036**
3d	17,738	0	**0.026**
spoštovan	11,177	0.019	**0.059**
2x	11,092	0	**0.016**
tožnik	9,909	0.008	**0.036**
odstotek	9,265	**0.515**	0.393
co2	9,090	0	**0.013**
amandma	8,992	0.007	**0.031**
hvala	8,954	0.106	**0.173**
1x	8,505	0	**0.012**
ekspr	8,373	0	**0.012**

Table 3: The first 15 lemmas with highest log-likelihood scores and their frequency per million words for the comparison of the old and new version of slWaC

As can be noted, most of these highest LL lemmas are more prominent in slWaC$_2$; only *"biti" (to be)* and *"odstotek" (percent)* are more frequent in slWaC$_1$. Furthermore, quite a few lemmas have frequency 0 in slWaC$_1$. This is indicative of a difference in annotation between the two corpora: as mentioned, the tokenisation module of ToTaLe had been somewhat improved lately, which is evidenced in the fact that strings, such as "m2" and "3d" were wrongly split into two tokens in slWaC$_1$ but are kept as one in slWaC$_2$. It is a characteristic of LL scores that they show such divergences, which should ideally be fixed, to arrive at uniform annotation of the resources.

4.1 Lemma comparison with slWaC$_1$

The motivation behind comparing the previous and current version of slWaC was primarily to investigate what kind of text types are better represented in the new (or old) version of the corpus. Apart from the already mentioned differences in tokenisation, slWaC$_2$ is more prominent in three types of lemmas (texts).

First, there are legal texts, characterised by lemmas such as *"člen" (article,) "odstavek" (paragraph)*, *"amandma" (amendment) "tožnik" (plaintiff)*, which come predominantly from governmental domains, e.g. for "člen" mostly from uradni-list.si (official gazette), dz-rs.si (parliament), sodisce.si (courts).

Second are texts that address the reader (or, say, parliamentary speaker) directly, such as *"spoštovan" (honoured)*, *"pozdravljen" (hello)*, *"hvala" (thank you)*. For "spoštovan", the most highly ranked domains are, again, the parliament, i.e. dz-rs.si, followed by vizita.si (medical help page of commercial POP.TV), delo.si (main Slovene daily newspaper), while "pozdravljen" and "hvala" come mostly from user forums. The corpus slWaC$_2$ is thus more representative in text-rich domains whose content changes rapidly and that contain user-generated content.

Third, the list contains two interesting "lemmas" with very high LL scores. The first is "ekspr" (only 19 in slWaC$_1$ but more than 9,000 in slWaC$_2$), which is the (badly tokenised) abbreviation "ekspr." meaning "expressive". It turns out that practically the only domain that uses this abbreviation is bos.zrc-sazu.si, i.e. the portal serving the monolingual Slovene dictionary SSKJ, which was newly harvested in slWaC$_2$. Similarly, the word "ino" (less than 500 in slWaC$_1$ but more than 7,000 in slWaC$_2$) turns out to be the historical form of *"in" (and)*. Practically the only domain containing this word (6,000x) is nl.ijs.si, which now hosts a large library of old Slovene books. The new slWaC thus contains some extensive new types of texts coming from previously unharvested domains or domains that have had large amounts of new content added.

Finally, it is worth mentioning that the first proper noun in slWaC$_2$ appears only at position 36 in the LL list, and is "bratušek" with almost 6,000 occurrences, referring to Alenka Bratušek, the former (2013 – 2014) PM of Slovenia.

It is also instructive to see which lemmas are now less specific against slWaC$_1$. Among function words, there is less conjunction "pa" used either as an informal version of *"in" (and)* or as an adversary conjunction *but*, and there is less of *"da" (that)*, used to introduce relative clauses. The drop in the frequency of the conjunction "pa" seems to have a link in the increase of the conjunction *"in" (and)* which now demonstrates more than 22 million occurrences. Significantly lower appearance of "da" can be explained by the fact that verbs such as *"dejati" (to say)*, *"poročati" (to report)*, *"pojasniti" (to explain)*, *"povedati" (to tell)*, *"sporočiti" (to communicate)*, and *"napovedati" (to predict)*, which are usually followed by the conjunction "da" are now much less used in slWaC$_2$. Those verbs are typical for news reporting and the drop in their usage indicates a drop in the proportion of news items in the corpus.

Most of the bottom part of the LL list, of course, consists of nouns and adjectives – and all of them again confirm that harvesting of texts for slWaC$_2$ was much less focused on news portals than for the previous one. Namely, as a previous frequency profiling of Gigafida and KRES shows [2] lemmas like *"odstotek" (percent)*; *"milijon" (milion)*, *"evro" (euro)*, *"dolar" (dollar)*, *"tolar" (former Slovene currency)*; *"predsednik" (president)*, *"premier" (prime minister)*, *"država" (state)*, *"minister" (minster)*; *"ameriški" (American)*, *"britanski" (British)*, *"hrvaški" (Croatian)*, *"nekdanji" (former)*, *"leto" (year)*, *"lani" (last year)*, and *"zdaj" (now)* all typically appear in daily newspapers (or, in our case, on news portals) reporting on interior and international affairs – and, as mentioned, we found all of them at the bottom of the LL list, indicating less news in slWaC$_2$ than in slWaC$_1$ and also the shifting of major news topics (for instance from Kosovo and Iraq).

4.2 Lemma comparison with KRES

With slWaC$_2$, as with Web corpora in general, it is an interesting question of how representative and balanced they are. The easiest approach towards an answer to this question is a comparison with "traditional" reference corpora, and such experiments have been already performed, e.g. between the British Web corpus ukWaC and BNC, the British National Corpus [1]. The comparisons have shown that while Web corpora are different from classical corpora, which contain mostly printed sources, the differences are in general not great and so they can function as modern-day reference corpora.

We made a comparison between slWaC$_2$ and KRES [13], the balanced reference corpus of Slovene with 100 million words. The comparison shows that, as with slWaC$_1$, some of the differences are due to the different linguistic analyses. As mentioned, slWaC$_2$ was processed with ToTaLe, while KRES used the Obeliks tokeniser, tagger and lemmatiser [7], and the two disagree in some lemmatisations, the most prominent being *"veliko/več" (much)*, *"mogoče/mogoč" (possible)*, *"edini/edin" (only)*, *"desni/desen" (right)*, *"levi/lev" (left)*, *"volitve/volitev" (elections)*, as well as some differences in tokenisation, e.g. "le-ta" and "d.o.o." as one token or three.

Real linguistic differences concern mostly two types of lemmas. The first are highly ranked non-content words such as *"pa, tudi, sicer, ter, naš" (but, also, otherwise, and, our)*, which most likely show the bias of slWaC$_2$ texts to antithetical and intensifying sentences, sentences with binding clause elements (adducing), and sentences which either (a) describe characteristics of the institution representing itself on the Web – "naši programi, naša spletna stran" (our programmes, our Web page); (b) establish a common communication circle [9, 6] – *"naša dežela, naši plezalci" (our country, our climbers)*, or (c) include readers into a text – *"naša duhovna rast, naša pot" (our spiritual growth, our path)*. The second are content lemmas, which fall into several groups: *"spleten" (Web)*, *"podjetje" (company)*, *"tekma, ekipa" (match, team)*, *"sistem, uporabnik, aplikacija" (system, user, application)*, and *"blog" (blog)*, i.e. slWaC$_2$ has more commercial, sports, and computer related texts, and, of course, text specific to the Web (blogs).

Conversely, KRES shows more lemmas to do with legal texts, such as *"član, odstavek, zakon" (article, paragraph, law)*, so that even with slWaC$_2$ having more texts of this type than slWaC$_1$, it still has much less than KRES.

KRES also has a specific group of lemmas, thematising a person in relation to another person, e.g. *"mama, oče, mož, žena" (mother, father, husband, wife)*, and verbs characteristic for interpersonal communication – *"vprašati, nasmehniti se, prikimati, zasmejati se" (to ask, to smile, to nod, to laugh)*. All these mostly come from fiction books in KRES. Two more specific lemmas are worth mentioning: *"tolar" (former Slovene currency)* shows that KRES, unlike slWaC$_2$, contains texts dating before 2007 (the changeover year to the euro in Slovenia), while "wallander", the hero of a series of detective novels, shows that KRES – at least in this instance – has too much text from a single source, here a book series.

4.3 Lemma comparison with Gigafida

Not surprisingly, a comparison between slWaC$_2$ and the Gigafida corpus showed rather similar results to the comparison between slWaC$_2$ and KRES. The top part of the list again contains content lemmas like *"spleten" (Web)*, *"aplikacija" (application)*, *"blog"*, *"uporabnik" (user)*, *"facebook"*, *"sistem" (system)*, etc., indicating slWaC$_2$ has more computer and Web related texts. However, the interesting part is the part where the two LL lists differ. First, it is obvious the Gigafida korpus has more sport related texts than KRES, therefore lemmas like *"tekma" (game)*, *"ekipa" (team)*, *"rezultat" (result)*, *"sezona" (season)*, *"trening" (training)* and *"zmaga" (victory)* are less prominent in slWaC$_2$and in KRES. The lemma *"podjetje" (company)* has a much lower LL score now as well, showing it is thematised in Gigafida in a larger proportion of texts than in KRES. Lemmas that are specific to slWaC$_2$ when we compare it to Gigafida (and not, when we compare it to KRES) are mostly non-content words, such as conjunctions *"in, ali, če" (and, or, if)*, personal pronouns *"jaz, ti" (I, you)*, and possessive personal pronouns *"moj, tvoj, vaš" (my, your$_{sg}$, your$_{pl}$)*, which show slWaC$_2$ contains more first and second person related contents most likely coming from user generated texts.

The lowest part of the LL list shows lemmas specific to Gigafida indicating Gigafida's bias towards news reporting texts thematising internal affairs, economy, and crime: *"predsednik" (president)*, *"minister" (minister)*, *"vlada" (government)*, *"občina" (municipality)*, *"prodati" (to sell)*, *"direktor" (manager)*, *"milijon" (million)*, and *"policist" (police officer)*, cf. [2].

4.4 Grammatical comparison with KRES

Apart from lemmas, it is also interesting to compare how the distribution of morphosyntactic categories of slWaC$_2$ differs from that of KRES. To this end we calculated six LL comparison scores, for uni-, bi- and trigrams of part-of-speech (PoS) and of complete morphosyntactic descriptions (MSDs).

The unigram PoS LL scores show that slWaC$_2$ has significantly more adjectives, unknown words, conjunctions, prepositions and particles, in this order. However, it has much less punctuation and numerals, and slightly less interjections. Especially with unknown words and punctuation the differences might be, at least partially, an artefact of different annotation programs. For the others, the results show that slWaC$_2$ tends more towards informal, user generated language (typical lemma for which is also *"lp"* meaning *"lep pozdrav" (best regards)* placed at position 20 in the LL list), although this conclusion is somewhat offset by the fact that it has less interjections. However, tagging interjections is notoriously imprecise, and the difference here might also be due to different taggers used. Conversely, KRES with its numerals shows a preponderance of newspaper texts, which tend to use lots of dates, times, amounts, and sports scores.

PoS bigrams again highlight the different annotation tools used. The most prominent combination in slWaC$_2$ is a numeral followed by an abbreviation, e.g. *"90 EUR, 206 kW, 298,80 m2"* but this difference is due to the fact that in slWaC$_2$ "EUR", "kW" etc. are treated as abbreviations, whereas they are common nouns in KRES. The same reasoning applies to combinations with punctuation. However, there are also legitimate combinations in the top scoring LL PoS bigrams: slWaC$_2$ has more noun + verb, adjective + noun and verb + adjective combinations, while KRES has more numeral + numeral, numeral + noun and verb + verb combinations. Scores for PoS trigrams give little new information: apart from annotation differences, the most prominent slWaC$_2$ combination is noun + noun + verb, which are mostly name + surname + predicate, e.g. *"Oto Pestner naredil"*, while the most prominent for KRES is a sequence of three numerals.

As for MSDs, the differences in unigrams in favour of slWaC$_2$ are greatest for the three unknown word types that KRES doesn't use (Xf: foreign word, Xp: program mistake and Xt: typo), followed by general adverbs in the positive degree, coordinating conjunctions, present tense first person auxiliary verb in the plural (*"smo"*) and animate common masculine singular noun in the accusative, i.e. the object of a sentence, e.g. *"otroka"*. Conversely, KRES has much more punctuation, digits, common masculine and feminine singular nouns in the nominative (i.e. subjects) and general adverbs in comparative and superlative degrees. Bigrams show that slWaC$_2$ has many more general adjective + common noun combinations in various genders and cases, while KRES has many more combinations with digits. The space of MSD trigrams is very large, and, if we discount the combinations appearing as a result of different annotations, does not show very interesting differences.

5 Conclusion

The paper presented a new version of the Slovene Web corpus, which is almost three times larger than its initial version and is made available through a powerful and freely accessible concordancer. During the construction process we focused on the content reductions obtained through near-duplicate removal, showing that both reductions to document and paragraph level remove a similar amount of content. We also compared the content of the slWaC$_2$ corpus to three other Slovene corpora (the slWaC$_1$ corpus, the balanced reference corpus KRES and the reference corpus Gigafida) with frequency profiling on lemmas and grammatical descriptions.

This comparison showed that the new version of the corpus has significantly more legal texts and specific text types, such as a dictionary and a library of historical books and (comparatively) less news. In the lemma comparison with KRES it has less legal texts but more user generated content and more commercial, sports, political and computer related texts. The comparison with Gigafida again showed slWaC$_2$ has more computer and Web related texts, while in this case sports and commercial news were no longer slWaC$_2$ specific. A larger proportion of several personal pronouns indicated a significant difference in the extent of the user generated content between the two corpora as well. The comparison of grammatical categories also shows a bias to informal writing and against newspaper items. But maybe the most surprising (although, in retrospect, quite logical) insight of the comparison using frequency profiling is that it is a very good tool to detect even slight differences in the processing pipelines used for the compared corpora, which then lead to significant differences in the (token, lemma and MSD) vocabularies.

There are several directions that our future work could take. First, by constructing the second version of two out of four existing Web corpora of South Slavic languages, two ideas have emerged: one is to build a multilingual corpus consisting of all South Slavic languages, and the second to develop a monitor corpus which would be automatically extended with new crawls in predefined time frames. The second direction is in the annotation of the corpus, where more effort should be invested in developing a gold standard processing pipeline, which could then be used to re-annotate the Slovene corpora in a unified manner. In addition, given that the Web contains a significant portion of user generated content containing non-standard language, the annotation pipeline should be extended by introducing a standardisation (normalisation) step on wordforms, similar to our approach to modernisation of historical Slovene words [16], which would then give better lemmas and MSDs, allowing for easier exploration of Web corpora.

As to the slWaC$_2$ functioning as a modern-day reference corpus of Slovene, the analysis showed considerable differences in the three corpora. In the future we therefore intend to supplement the results of the lemma comparison with the results of the topic modelling method [3, 17, 2]. From the assembled data of both methods we will be able to estimate more precisely which texts each corpus contains and, perhaps even more importantly, which texts each corpus misses. We believe the building of the next generation reference corpus of Slovene could in this way greatly benefit from the slWaC$_2$ corpus – its contents as well as its construction methodology.

References

[1] Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. The WaCky wide Web: a collection of very large linguistically processed Web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226, 2009.

[2] Nataša Logar Berginc and Nikola Ljubešić. Gigafida in slWaC: tematska primerjava. *Slovenščina 2.0*, 1(1):78–110, 2013.

[3] Michael I. Jordan David M. Blei, Andrew Y. Ng and John Lafferty. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[4] Tomaž Erjavec, Camelia Ignat, Bruno Pouliquen, and Ralf Steinberger. Massive multilingual corpus compilation: Acquis Communautaire and ToTaLe. *Archives of Control Sciences*, 15(3):253–264, 2005.

[5] Tomaž Erjavec and Simon Krek. The JOS Morphosyntactically Tagged Corpus of Slovene. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, 2008.

[6] Monika Kalin Golob and Nataša Logar. Prostor v poročevalskem skupnem sporočanjskem krogu. *Slavistična revija*, 62(3):363–373, 2014.

[7] Miha Grčar, Simon Krek, and Kaja Dobrovoljc. Obeliks: statistični oblikoskladenjski označevalnik in lematizator za slovenski jezik. In *Zbornik Osme konference Jezikovne tehnologije*, Ljubljana, 2012. Jožef Stefan Institute.

[8] Emiliano Guevara. NoWaC: A Large Web-based Corpus for Norwegian. In *Proceedings of the NAACL HLT 2010 Sixth Web As Corpus Workshop*, WAC-6 '10, pages 1–7, 2010.

[9] Tomo Korošec. *Stilistika slovenskega poročevalstva*. Kmečki glas, Ljubljana, 1998.

[10] Nikola Ljubešić and Antonio Toral. caWaC - a Web Corpus of Catalan and its Application to Language Modeling and Machine Translation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, may 2014. European Language Resources Association (ELRA).

[11] Nikola Ljubešić. {bs,hr,sr}WaC: Web corpora of Bosnian, Croatian and Serbian. In *Proceedings of the WAC-9 Workshop*, 2014.

[12] Nikola Ljubešić and Tomaž Erjavec. hrWaC and slWac: Compiling Web Corpora for Croatian and Slovene. In Ivan Habernal and Václav Matousek, editors, *Text, Speech and Dialogue - 14th International Conference, TSD 2011, Pilsen, Czech Republic, September 1-5, 2011. Proceedings*, Lecture Notes in Computer Science, pages 395–402. Springer, 2011.

[13] Nataša Logar, Miha Grčar, Marko Brakus, Tomaž Erjavec, Špela Arhar Holdt, and Simon Krek. *Korpusi slovenskega jezika Gigafida, KRES, ccGigafida in ccKRES: gradnja, vsebina, uporaba*. Zbirka Sporazumevanje. Trojina, zavod za uporabno slovenistiko: Fakulteta za družbene vede, Ljubljana, 2012.

[14] Paul Rayson and Roger Garside. Comparing Corpora Using Frequency Profiling. In *Proceedings of the Workshop on Comparing Corpora*, pages 1–6. Association for Computational Linguistics, 2000.

[15] Pavel Rychlỳ. Manatee/bonito – a modular corpus manager. *1st Workshop on Recent Advances in Slavonic Natural Language Processing*, pages 65–70, 2007.

[16] Yves Scherrer and Tomaž Erjavec. Modernizing historical Slovene words with character-based SMT. In *BSNLP 2013 - 4th Biennial Workshop on Balto-Slavic Natural Language Processing*, Sofia, Bulgaria, 2013.

[17] Serge Sharoff. Analysing Similarities and Differences between Corpora. In *Proceedings of the Seventh Conference on Language Technologies*, pages 5–11, Ljubljana, 2010. Jožef Stefan Institute.

[18] Drahomíra Spoustová, Miroslav Spousta, and Pavel Pecina. Building a Web Corpus of Czech. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, 2010. European Language Resources Association (ELRA).

[19] Vít Suchomel and Jan Pomikálek. Efficient Web Crawling for Large Text Corpora. In Serge Sharoff Adam Kilgarriff, editor, *Proceedings of the seventh Web as Corpus Workshop (WAC7)*, pages 39–43, Lyon, 2012.

A Rule-Based System for Automatic De-identification of Medical Narrative Texts

Jelena Jaćimović[1,2], Cvetana Krstev[1] and Drago Jelovac[2]
[1]University of Belgrade, Faculty of Philology, Studentski trg 3, 11000 Belgrade, Serbia
[2]University of Belgrade, School of Dental Medicine, Dr. Subotića 8, 11000 Belgrade, Serbia
E-mail: jjacimovic@rcub.bg.ac.rs, cvetana@matf.bg.ac.rs, drago.jelovac@stomf.bg.ac.rs

Keywords: named entity recognition, finite-state transducers, rule-based system, de-identification, medical narrative texts, Serbian

Received: May 7, 2014

This paper presents an automatic de-identification system for Serbian, based on the adaptation of the existing rule-based named entity recognition system. Built on a finite-state methodology and lexical resources, the system is designed to detect and replace all the explicit personal protected health information present in the medical narrative texts, while still preserving all the relevant medical concepts. The results of a preliminary evaluation demonstrate the usefulness of this method both in preserving patient privacy and the de-identified document interoperability.

Povzetek: Razvit je nov sistem za de-identifikacijo besedil v srbskem jeziku.

1 Introduction

Current advances in health information technology enable health care providers and organizations to automate most aspects of the patient care management, facilitating collection, storage and usage of patient information. Such information, stored in the form of electronic medical records (EMRs), represents accurate and comprehensive clinical data valuable as a vital resource for secondary uses such as quality improvement, research, and teaching. Besides the vast useful information, narrative clinical texts of the EMR also include many items of patient identifying information. For both ethical and legal reasons, when confidential clinical data are shared and used for research purposes, it is necessary to protect patient privacy and remove patient-specific identifiers through a process of the de-identification.

De-identification is focused on detecting and removing/modifying all explicit personal Protected Health Information (PHI) present in medical or other records, while still preserving all the medically relevant information about the patient. Various standards and regulations for health data protection define multiple directions to achieve de-identification, but the most frequently referenced regulation is the US Health Information Portability and Accountability Act (HIPAA) [1]. According to the HIPAA "Safe Harbor" approach, clinical records are considered de-identified when 18 categories (17 textual and one regarding images) of PHI are removed, and the remaining information cannot be used alone or in combination with other information to identify an individual. These PHI categories include names, geographic locations, elements of dates (except year), telephone and fax numbers, medical record numbers or any other unique identifying numbers, among others. Since the manual removal of PHI by medical professionals proved to be prohibitively time-consuming, tedious, costly and unreliable [2, 3, 4], extracting PHI requires more reliable, faster and cheaper automatic de-identification systems based on the Natural Language Processing (NLP) methods [5].

The extraction of PHI can be viewed as a Named Entity Recognition (NER) problem applied in the medical domain for de-identification [6]. However, even though both the traditional NER and de-identification involve automatic recognition of particular phrases in text (persons, organizations, locations, dates, etc.), de-identification differs importantly from the traditional NER [7]. In contrast to the general NER focused on newspaper texts, de-identification deals with the clinical narratives characterized by fragmented and incomplete utterances, the lack of punctuation marks and formatting, many spelling and grammatical errors, as well as domain specific terminology and abbreviations. Since de-identification is the first step towards identification and extraction of other relevant clinical information, it is extremely important to overcome the problem of significantly large number of eponyms and other non-PHI erroneously categorized as PHI. For instance, the anatomic locations, devices, diseases and procedures could be erroneously recognized as PHI and removed (e.g. *"The Zvezdara method"*[1] vs. *Clinical Center "Zvezdara"*), reducing the usability and the overall meaning of clinical notes, and thus the accuracy of

[1] The original surgical 2-step arteriovenous loop graft procedure developed in the Clinical Center "Zvezdara", Belgrade, Serbia. *Zvezdara* is a municipality of Belgrade.

subsequent automatic processes performed on the de-identified documents.

In this paper we introduce our automatic clinical narrative text de-identification system, based on the adaptation of the existing rule-based NER system for Serbian. The aim of this study is to evaluate the accuracy of PHI removal and replacement while preserving all the medically relevant information about a patient and keeping the resulting de-identified document usable for subsequent information extraction processes.

2 Related work

Over the past twenty years, various text de-identification approaches have been developed, but relatively few published reports are focused only on the unstructured medical data. Extensive review of recent research in automatic de-identification of narrative medical texts is given in [5]. However, most of them are highly specialized for specific document types or a subset of identifiers. Regarding the general nature of applied de-identification methods, the majority of the systems used only one or two specific clinical document types (pathology reports, discharge summaries or nursing progress notes) for the evaluation [3, 8, 9, 10], while only a few of them were evaluated on a larger scale, with a more heterogeneous document corpus [11, 12, 13, 14]. The selection of targeted PHI varied from patient names only [12] to all 17 textual HIPAA PHI categories [3, 7, 15, 16, 17, 18, 19], or even everything but valid medical concepts [20, 21].

The de-identification approaches applied in medical domain are mostly classified into the rule-based or machine learning methods, while some hybrid approaches [14] efficiently take advantage of both previous methods. The rule-based methods [3, 15, 17, 19, 21] make the use of dictionaries and hand-crafted rules to identify mentions of PHI, with no annotated training data. Although these systems are often characterized with the limited generalizability that depends on the quality of the patterns and rules, they can be easily and quickly modified by adding rules, dictionary terms or regular expressions in order to improve the overall performance [22]. On the other hand, the machine-learning methods [7, 8, 9, 16, 18, 23], proved to be more easily generalized, automatically learn from training examples to detect and predict PHI. However, these methods require large amounts of annotated data and the adaptation of the system might be difficult due to the often unpredictable effects of a change. Extensive review of the published strategies and techniques specifically developed for de-identification of EMRs is given in [24]. In 2006, within the Informatics for Integrating Biology and the Bedside (i2b2) project and organized de-identification challenge, a small annotated corpus of hospital discharge summaries were shared among the interested participants, providing the basis for the system development and evaluation. Detailed overview and evaluation of the state-of-the-art systems that participated in the i2b2 de-identification challenge is given in [25].

Aside from systems specifically designed for the purpose of de-identification, some NER tools trained on newspaper texts also obtained respectable performance with certain PHI categories [7, 26].

3 Data and methods

This section provides an overview of our rule-based de-identification approach for narrative medical texts.

3.1 Training and text corpus

The training corpus for our system development consisted of 200 randomly selected documents from different specialties, generated at three Serbian medical centers. They included discharge summaries (50), clinical notes (50) and medical expertise (100), with a total word count of 143,378. The discharge summaries and clinical notes are unstructured free text typed by the physicians at the conclusion of a hospital stay or series of treatments, including observations about the patient's medical history, his/her current physical state, the therapy administered, laboratory test results, the diagnostic findings, recommendations on discharge and other information about the patient state. Medical expertise documents were oversampled because of their richness in the PHI items.

Main characteristics of medical narratives were confirmed in our corpus: fragmented and incomplete utterances and lack of punctuation marks and formatting. Moreover, as these documents are usually written in a great hurry there is also an unusual number of spelling, orthographic and typographic errors, much larger than in, for instance, newspaper texts from the Web. For the moment, we have taken these documents as they are and we are not attempting to correct them. In some particular situations we are able to guess the intended meaning, as will be explained in the next section.

3.2 The NER system

The primary resources for natural language processing of Serbian consist of lexical resources and local grammars developed using the finite-state methodology as described in [27, 28]. For development and application of these resources the Unitex corpus processing system is used [29]. Among general resources used for NER task are the morphological e-dictionaries, covering both general lexica and proper names, as well as simple words and compounds, including not only entries collected from traditional sources, but also entries extracted from the processed texts [30]. Besides e-dictionaries, for the recognition and morphosyntactic tagging of open classes of simple words and compounds generally not found in dictionaries, the dictionary graphs in the form of finite-state transducers (FSTs) are used. Due to the high level of complexity and ambiguity of named entities, the additional resources for NER were developed [31]. The Serbian NER system is organized as a cascade of FSTs – CasSys [32], integrated in the Unitex corpus processor. Each FST in a cascade modifies a piece of text by replacing it with a lexical tag that can be used in subsequent FSTs. For instance, in a sequence *Dom* *zdravlja "Milutin Ivković"* 'Health Center 'Milutin Ivković" first a full name 'Milutin Ivković' is recognized and tagged {Milutin Ivković,.NE+persName+full:ms1v}, and then a subsequent transducer in the cascade uses this information to appropriately recognize and tag the full organization name that can also be subsequently used (see Figure 1 and Example (1)).

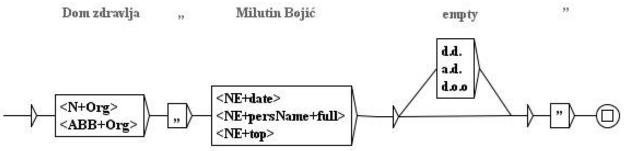

Figure 1. A path in a cascade graph that uses already recognized NEs to recognize organization names.

(1){Dom zdravlja "\{Milutin Ivković\, \.NE\+persName \+full\\:s1v\}",.NE+org:1sq:4sq}

Serbian NER system recognizes a full range of traditional named entity types:

- Amount expressions – count, percentage, measurements and currency expressions;
- Time expressions – absolute and relative dates and times of day (fixed and periods), durations and sets of recurring times;
- Personal names – full names, parts of names (first name only, last name only), roles and functions of persons;
- Geopolitical names – names of states, settlements, regions, hydronyms and oronyms;
- Urban names – at this moment only city areas and addresses are recognized.

For the purpose of PHI de-identification not all of these NEs are of interest. For instance, amount expressions should not be de-identified, and roles or functions need not be de-identified. However, we chose not to exclude them from the recognition for two reasons: first, if they are recognized correctly that may prevent some false recognition and second, even if they are not of interest for this specific task they may help in recognition of some NEs that are of interest. For instance in Example (2) a name is erroneously typed (both the first and the last name are incorrect) but due to a correct recognition of a person's function the name is also recognized.

(2) *prof. dr sci Drangan Jorvanović, specijalista za stomatološku protetiku i ortopediju* 'Prof. PhD Drangan Jorvanović, a specialist for Prosthetic Dentistry and Orthodontics'

The finite-state transducers used in the NER cascade beside general and specific e-dictionaries, as explained before, use local grammars that model various triggers and NE contexts, such as:

- The use of upper-case letters – for personal names, geopolitical names, organizations, etc.;
- The sentence boundaries – to resolve ambiguous cases where there is not enough other context;
- Trigger words – for instance, *reka* 'river', *grad* 'city' and similar can be used to recognize geopolitical names that are otherwise ambiguous;
- Other type of the context – for instance, a punctuation mark following a country name that coincides with a relational adjective[2] signals that it is more likely a country name than an adjective;
- Other NEs – for instance, an ambiguous city name can be confirmed if it occurs in a list of already recognized NEs representing cities. Also, a five digit number that precedes a name of a city (already recognized) is tagged as a postal code (as used in Serbia).
- Grammatical information – this information is used to impose the obligatory agreement in the case (sometimes also the gender and the number) between the parts of a NE. For instance, in *…istakao je gradonačelnik Londona Boris Džonson…* '…stressed Mayor of London Boris Johnson…' *Londona* can be falsely added to the person's name (because *London* is also a surname) if grammatical information was not taken into consideration (*Londona* is in the genitive case, while *Boris* and *Džonson* are in the nominative case). This is enabled by grammatical information that is part of NE lexical tags (see Example (1)).

3.3 The PHI de-identification

We used our training corpus to create and adapt patterns that will capture the characteristics of PHI. Through the corpus analysis we found that, out of 18 HIPAA PHI categories, only eight appeared in our data. Since there is no annotation standard for PHI tagging, we collapsed some of the HIPAA categories into one (telephone and fax numbers, medical record numbers or any other unique identifying numbers). In order to maximize patient confidentiality, we adopted a more conservative approach, considering countries and organizations as PHI. For the purposes of this study, we defined the resulting PHI categories as follows:

- Persons (*pers*) – refers to all personal names; includes first, middle and/or last names of patients and their relatives, doctors, judges, witnesses, etc.;
- Dates (*date*) – includes all elements of dates except year and any mention of age information for patients over 89 years of age; according to HIPAA, the age over 89 should be collected under one category 90/120;
- Geographic locations (*top*) – includes countries, cities, parts of cities (like municipalities), postal codes;
- Organizations (*org*) – hospitals and other organizations (like courts);
- Numbers (*num*) – refers to any combination of numbers, letters and special characters

[2] In Serbian many country names coincide with relational adjectives of feminine gender: *Norveška* 'Norway' and *norveška* 'Norwegian'.

representing telephone/fax numbers, medical record numbers, vehicle identifiers and serial numbers, any other unique identifying numbers;

- Addresses (*adrese*) - street addresses.

The processing usually starts with a text having undergone a sentence segmentation, tokenization, part-of-speech tagging and morphological analysis. After general-purpose lexical resources are used to tag the text with lemmas, grammatical categories and semantic features, the FST cascade is applied, recognizing persons, functions, organizations, locations, amounts, temporal expressions, etc.

Since medical narratives have specific characteristics, the primary issue of date's recognition arose and we added a small cascade of FSTs prior to detection of the sentences. For the de-identification task and the processing of medical data, we performed the adjustments of the temporal expressions FSTs, in order to cover only those temporal expressions that should be treated as PHI. We also developed new patterns for the identification of different diagnostic codes present in training documents that could be misinterpreted as an identifier and then erroneously masked. Being applied as first in the cascade, this FST produces lexical tags denoting non-PHI category of the diagnostic codes, bringing the precision and accuracy up to an acceptable level in order to prevent loss of clinical information in the de-identification process.

Lexical tags produced by FSTs (see Example (3)), even though the most convenient for the use of subsequent FSTs in the cascade, are not useful for other applications and at the end are converted to the XML tags (Example (4)).

(3) {Beogradu,.NE+top+gr:s7q}
(4) <top.gr>Beogradu</top.gr>

The de-identification can be performed in several ways: PHI that needs to be de-identified can be replaced by a tag denoting its corresponding category, with a surrogate text, or both. We have chosen the latter approach. Moreover, since we are dealing with the narrative texts as a result we want to obtain a narrative text as well. To that end, the surrogate text is chosen to agree in case, gender and number with the PHI it replaces (if applicable). Again, such a replacement is enabled by grammatical information associated with some NE types (personal names, organization names, locations, etc.). For instance, recognized geographic name (+top) of the city (+gr) in Example (3) will be replaced by the surrogate text with the same values of the grammatical categories (Example (5)).

(5) <top.gr PHI="yes">Kamengradu</top.gr>

At this moment, our system does not keep the internal structure of the numbers PHI category (*num*), and all the PHI numbers are simply replaced by placeholder characters X. Regarding the temporal information, only the month and day portion of date expressions are considered PHI. According to HIPAA, the years are excluded from this category, being important features of the clinical context. In order to preserve the existing interval in days between two events in the text or the duration of specific symptoms, all dates were replaced by a shifted date that is consistent throughout all the de-identified documents.

3.4 An example

In this subsection we will give an example taken from the part of the test corpus containing medical expertise. The part of one note is given in Example (6).[3] The same expertise after the de-identification and tagging is given in Example (7).[4]

(6)
Naš broj 23/246
OPŠTINSKI SUD - Istražni sudija G-đa Jovana Jovanović-
Vašim zahtevom u predmetu TR 123/01 od 23.07.2007. god. zatražili ste od Komisije lekara veštaka Medicinskog fakulteta Univerziteta u Kragujevcu sudskomedicinsko veštacenje o vrsti i težini telesnih povreda koje je dana 4.02. 2007. god. zadobio Petrović Dragan iz Jagodine.
…
PODACI
1. Pri pregledu na Medicinskom fakultetu u Kragujevcu, obavljenom dana 12.02.2007. god. od strane članova Komisije lekara veštaka Medicinskog fakulteta Univerziteta u Kragujevcu, Dragan, Miroslava, Petrović navodi: rođen je 14. 01. 1956. god. u Jagodini, živi u Jagodini, ul. Savska br. 7, po zanimanju pekar, broj lične karte 1234567, MUP Jagodina . Amanestički navodi operaciju kolena marta 2000. god., negira postojanje oboljenja. Dana 4,02. 2007. god. oko 12,30 h, na sportskom terenu došlo je do fizičkog obračuna između Dragana i njegovog poznanika.
…
NALAZ
1. U izveštaju Dr Petra Dragića, specijaliste za otorinolaringologiju, Zdravstvenog centra "Milutin Ivković", iz Jagodine, na ime Petrović Dragana, od 4.02.07. god., navedeno je sledeće: "Povređen u tuči od strane poznatog lica. Svest nije gubio. Dg. Fractura dentis incisiv"
…

'Our number 23/246
Municipal court - Judge Mrs Jovana Jovanović -
In your request in the case TR 123/01 from 23/07/2007 you asked for a medico-legal expertise on the type and

[3] This example looks exactly as the original – however, for the purpose of protecting the personal data we have manually replaced all the personal information with some "real world" data.

[4] We wanted to avoid the introduction of some real people names and real location names in the de-identified texts. Instead we used names: *Barni Kamenko* (Barney Rubble), *Vilma Kremenko* (Vilma Flintstone), *Kamengrad* (Bedrock), Serbian names for the characters from the sitcom *The Flinstones*, created by Hanna-Barbera Productions, Inc.

gravity of bodily injuries inflicted on Petrović Dragan from Jagodina on 4/02/2007 from the Commission of the Faculty of Medical Sciences University of Kragujevac's medical experts.
…
DATA
1. Upon the examination performed in Faculty of Medical Sciences of Kragujevac, conducted on 12/02/2007 by members of the Commission of the Faculty of Medical Sciences University of Kragujevac's medical experts, Dragan, Miroslava, Petrović states: born on 14/ 01/1956 in Jagodina, lives in Jagodina, Savska Street 7, a baker by profession, ID number 1234567, MIA Jagodina. Anamnestic states the knee surgery performed on March 2000, negates the existence of a disease. On 4/02/2007 around 12:30 PM, on the sports field it came to a physical confrontation between Dragan and his acquaintance.
…
FINDING
1. In the medical report of Zoran Dragić, MD, specialised in otorhinolaryngology, Medical Centre "Milutin Ivković", from Jagodina, on Petrović Dragan's name, from 4/02/2007, the following was stated: "He was injured in a fight by an acquaintance. He didn't lose his consciousness. Dg. Fractura dentis incisiv. "
…'

(7)
Naš <number PHI="yes">XXXX</number>
<org PHI="yes">SUD</org> - <pers><role>Istražni sudija gospođa</role> <persName.full PHI="yes">Vilma Kremenko</persName.full></pers>- Vašim zahtevom u predmetu <number PHI="yes">XXXX</number> od <date PHI="yes">28.12.2007.</date> zatražili ste od <org PHI="yes">Komisije</org> <org PHI="yes">fakulteta</org> <org PHI="yes">Univerziteta</org> sudskomedicinsko veštacenje o vrsti i težini telesnih povreda koje je dana <date PHI="yes">09.07.2007.</date> zadobio <persName.full PHI="yes">Barni Kamenko</persName.full> iz <top.gr PHI="yes">Kamengrada</top.gr>.
…
PODACI
1.{S} Pri pregledu na <org PHI="yes">fakultetu</org>, obavljenom dana <date PHI="yes">17.07.2007.</date> od strane članova <org PHI="yes">Komisije</org> <org PHI="yes">fakulteta</org> <org PHI="yes">Univerziteta</org>, <persName.full PHI="yes">Barni Kamenko</persName.full> navodi: rođen je <date PHI="yes">19.06.1956.</date> u <top.gr PHI="yes">Kamengradu</top.gr>, živi u <top.gr PHI="yes">Kamengradu</top.gr>, <adresa PHI="yes">ul. Kamenolomska br. 6a</adresa>, po zanimanju pekar, broj lične karte <number PHI="yes">XXXX</number>, <org>MUP<top.gr PHI="yes">Kamengrad</top.gr></org> .{S} Amanestički navodi operaciju kolena <date PHI="yes">avgusta 2000.</date>, negira postojanje oboljenja.{S} Dana <date PHI="yes">09.07.2007.</date> oko 12,30 h, na sportskom terenu došlo je do fizičkog obračuna između **Dragana** i njegovog poznanika.
…
NALAZ
1.{S} U izveštaju <pers><persName.full PHI="yes">Barnija Kamenka</persName.full><role>, specijaliste za otorinolaringologiju</role></pers>, <org PHI="yes">centra</org>, iz <top.gr PHI="yes">Kamengrada</top.gr>, na ime <persName.full><persName.full PHI="yes">**Vilma Kremenko**</persName.full></persName.full>, od <date PHI="yes">09.07.07.</date>, navedeno je sledeće:"....{S} Povređen u tuči od strane poznatog lica.{S} Svest nije gubio.{S} Dg.{S} Fractura dentis incisiv."
…

This example demonstrates our de-identification approach. Each detected PHI was enclosed in XML tags indicating its corresponding category, with the PHI attribute value set to "yes". Note that all dates were shifted into the future by the same amount. Information specific to the hospitals and other organizations was replaced by a generalized data with the same organizational hierarchy. For instance, a sequence of hierarchical organization names *Komisija lekara veštaka Medicinskog fakulteta Univerziteta u Kragujevcu* 'the Commision of medical experts of the Faculty of Medical Sciencies of the University of Kragujevac' is replaced by *Komisija fakulteta Univerziteta* 'a Commission of a faculty of a University'. Some personal data remained: the occurrence of the first name of the patient. Also, the replacement text was not always correct: the male patient's name was once replaced by the female name because the original occurrence was ambiguous and could be interpreted both as a masculine name *Petrović Dragan* (in the genitive case) and a feminine name *Petrović Dragana* (in the nominative case). Our system has randomly chosen the feminine name. These occurrences are bolded and underlined in Example (7).

4 Evaluation results

The previously described system for the automatic de-identification has been evaluated on a set of 100 randomly selected documents (total word count of 35,822), consisting of discharge summaries (60), clinical notes (27) and medical expertise (13). These chosen texts were not used in the system development and presented completely unseen material containing many occurrences of PHI. Details about the PHI distribution within the test corpus can be found in Table 1.

The performance has been evaluated with respect to recognition, bracketing and replacement of PHI. For that reason, a new attribute 'check' has been added to each XML tag. Possible values of this attribute were the following:

OK – PHI was correctly recognized, full extent was correctly determined, replacement was correctly assigned;

UOK - UOK1 (PHI type was correctly recognized, but full extent was not correctly determined, some part of PHI was revealed); UOK2 (PHI type was not

PHI/Document type	Cinical reports	Discharge summaries	Medical expertise	Total
pers	52	254	407	713
top	32	219	109	360
org	62	164	242	468
num	20	61	90	171
date	65	133	267	465
adrese	0	64	10	74
Total	231	895	1125	2251

Table 1: The PHI distribution considering document type.

correctly determined, but the full extent was correctly determined, PHI successfully masked);

NOK – an utterance tagged falsely as PHI and de-identified;

MISS – PHI was not recognized;

MISS/E – PHI was not recognized because of the incorrect input.

In some cases when it was not so easy to decide which is the most appropriate value for the 'check' attribute (e.g. personal name as a name of an organization), we always treated as correct, for example, a personal name tag even though the utterance belonged to organization category.

We report the results of the evaluation using the traditional performance measures: precision (positive predictive value), recall (sensitivity) and *F*-measure (harmonic mean of recall and precision). These measures are calculated at the phrase level, considering the entire PHI annotation as the unit of evaluation.

The harmonic mean of recall and precision is calculated in two ways, using the strict and relaxed criteria. With the strict criteria we consider as true positives only fully correctly recognized and de-identified PHI and as false negatives all PHI that were not recognized and de-identified, regardless of the reasons (including the incorrect input). With the relaxed criteria we consider as true positives all correctly recognized and de-identified PHI including partial recognition and false type attribution, and as false negatives all PHI that were not recognized and de-identified if the input was correct (see Table 2).

The overall evaluation of the system is presented in Table 3 and Table 4.

	1. Strict criteria	2. Relaxed criteria
TP	OK	OK+UOK
FP	NOK+UOK	NOK
FN	MISS+MISS/E	MISS
P	OK/(OK+NOK+UOK)	(OK+UOK)/(OK+NOK+UOK)
R	OK/(OK+MISS+MISS/E)	(OK+UOK)/(OK+UOK+MISS)

Table 2. Calculation using strict and relaxed criteria: TP (true positive), FP (false positive), FN (false negative), P (Precision), R (Recall).

PHI	OK	UOK1	UOK2	MISS	MISS/E	NOK
pers	634	12	47	15	5	30
top	337	0	0	14	9	5
org	434	0	0	28	6	6
num	132	2	0	36	1	8
date	455	4	0	1	5	7
adrese	63	0	0	1	10	2
Total	2055	18	47	95	36	58

Table 3. Evaluation data.

PHI	Precision (p1)	Recall (r1)	*F1*-measure	Precision (p2)	Recall (r2)	*F2*-measure
pers	0.88	0.97	0.92	0.96	0.98	0.97
top	0.99	0.94	0.96	0.99	0.96	0.97
org	0.99	0.93	0.96	0.99	0.94	0.96
num	0.93	0.78	0.85	0.94	0.79	0.86
date	0.98	0.99	0.98	0.98	1.00	0.99
adrese	0.97	0.85	0.91	0.97	0.98	0.98
Total	**0.94**	**0.94**	**0.94**	**0.97**	**0.96**	**0.97**

Table 4. Performance measures for PHI de-identification by applying the strict criteria (1) and the relaxed criteria. (2)

Besides the traditional *F*-measure, evaluation is also performed using the Slot Error Rate (SER) [33]. As a simple error measure, the SER equally weights different types of error directly, enabling the comparison of all systems against the fixed base. The SER is equal to the sum of the three types of errors — substitutions (UOK1, UOK2), deletions (MISS, MISS/E), and insertions (NOK) — divided by the total number of PHI in the reference corpus (Formula (1)).

(1)

$$SER = \frac{NOK + MISS + MISS/E + UOK1 + UOK2}{OK + UOK1 + UOK2 + MISS + MISS/E}$$

In measuring the accuracy of the de-identification system, the extent of PHI (UOK1), missed PHI (MISS, MISS/E) as well as entities falsely tagged as PHI (NOK), should be taken into consideration as equally weight separate error slots. In that way, unlike the more relaxed *F*-measure, the SER of 11.3% stresses out the importance of the errors that affect revealing of PHI to a greater extent.

5 Discussion

Clinical records are considered de-identified when, after removal of PHI, the remaining information cannot be used alone or in combination with other information to identify an individual. Nevertheless, even though PHI is removed, there is a concern that de-identified medical documents could potentially be re-identified i.e. that it is difficult but still possible to reestablish the link between the individual and his/her data [24]. In the context of de-identification each PHI category is treated differently. There are obvious identifiers (e.g. name, telephone number, home address…) as well as quasi-identifiers that can play an important role in indirect re-identification, such as dates, locations, race and gender [34]. In some cases, more than one identifying variable is needed to identify an individual uniquely. For example, sex and year of birth combined with the disease name (it might be some rare disease) could be used for indirect re-identification. However, some PHI categories, such as ages over 89, geographical locations, hospitals and other organizations are most frequently ignored by the existing de-identification systems [5]. Even though according to the most frequently referenced regulation HIPAA, states and organizations are not considered as PHI, we adopted a more conservative approach, considering them as variables that could be used for re-identification.

We found that our NER system could be modified to work on medical narratives for de-identification purposes. However, certain modifications were necessary in order to preserve relevant clinical information. Previous evaluation results showed that Serbian NER system gives priority to precision over the recall [30], and the recall rate had to be improved for the de-identification task.

An error analysis shows that every correctly recognized PHI was correctly de-identified. The main source of errors were missed PHI, resulting in the information disclosure. The most missed PHI were numbers and organizations not included in our pattern rules and dictionaries, while fewer than 6% of errors resulted in the revealing of the most sensitive category i.e. person names. Another source of errors that could cause PHI exposure was wrongly determined PHI extent (4.72% of total errors). Fewer than 20% of errors were examples tagged with an incorrect PHI category which may only reduce the readability of the resulting de-identified text without exposing PHI. Since one of the main goals is to preserve medically relevant information, it is important to pay special attention to false positives, which represented 22.83% out of total errors. For *pers*, a majority of false positives were diseases and procedures names.

Our automatic de-identification system achieved very competitive precision and recall rate, showing the overall *F1*-measure of 0.94 (Table 4). High performance was achieved for most PHI types, except for numbers. The highest precision of 0.99 was reached for geographic locations and organizations, followed by dates, addresses and numbers. When partially recognized and wrongly tagged personal names are treated as true positives, the precision of their de-identification is better. With respect to the recall, the most important measure for de-identification, dates have the highest rate. Beside dates, almost all PHI categories showed high sensitivity rating from 0.99 to 0.93. The lowest recall rate for numbers (0.78) and addresses (0.85) suggests that rules for corresponding categories have to be improved. In terms of recall, especially dates and personal names, we may say that our de-identification is sufficient to guarantee high patient privacy, with achieved competitive precision and preserved document usefulness for subsequent applications.

6 Conclusion

In this paper, we presented the automatic text de-identification system for medical narrative texts, based on the adaptation of the existing rule-based NER system for Serbian. We have also produced the first versions of de-identified medical corpus that could be useful to the research community interested in both analysing different medical phenomena and producing a machine-learning automatic de-identification system for Serbian.

Even though the evaluation of the presented system is conducted on a relatively small set of documents, we have collected the heterogeneous corpus, consisting of different document types belonging to various medical specialties and institutions. The results of this preliminary evaluation are very promising, indicating that our adapted NER system can achieve high performance on the de-identification task. However, there is still much to be done.

In the future, we plan to focus on improving our strategies, such as completing the existing and adding the new patterns covering the broader formats of PHI (email addresses, URLs, IP address numbers) and the disambiguation of clinical eponyms and abbreviations. Finally, we intent to measure the impact of the de-identification through the subsequent natural language processing task of medical concepts' recognition.

References

[1] Health Insurance Portability and Accountability Act. P.L. 104-191, 42 USC. 1996.

[2] Douglass, M., Clifford, G. D., Reisner, A., Moody, G. B., Mark, R. G. 2004. Computer-assisted de-identification of free text in the MIMIC II database. Computers in Cardiology, 31:341-344.

[3] Neamatullah, I., Douglass, M. M., Lehman, L.-W. H., Reisner, A., Villarroel, M., Long, W. J., Szolovits, P., Moody, G. B., Mark, R. G. & Clifford, G. D. 2008. Automated de-identification of free-text medical records. BMC Medical Informatics and Decision Making, 8:32.

[4] Deleger, L., Molnar, K., Savova, G., Xia, F., Lingren, T., Li, Q., Marsolo, K., Jegga, A., Kaiser, M., Stoutenborough, L. & Solti, I. 2013. Large-

scale evaluation of automated clinical note de-identification and its impact on information extraction. Journal of the American Medical Informatics Association, 20:84-94.

[5] Meystre, S. M., Friedlin, F. J., South, B. R., Shen, S. & Samore, M. H. 2010. Automatic de-identification of textual documents in the electronic health record: a review of recent research. BMC Medical Research Methodology, 10:70.

[6] Nadeau, D. & Sekine, S. 2007. A survey of named entity recognition and classification. Lingvisticae Investigationes, 30:3-26.

[7] Wellner, B., Huyck, M., Mardis, S., Aberdeen, J., Morgan, A., Peshkin, L., Yeh, A., Hitzeman, J. & Hirschman, L. 2007. Rapidly retargetable approaches to de-identification in medical records. Journal of the American Medical Informatics Association, 14**:**564-573.

[8] Gardner, J. & Xiong, L. 2008. HIDE: An integrated system for health information DE-identification. In: Proceedings of the 21st IEEE International Symposium on Computer-Based Medical Systems. 254-259.

[9] Uzuner, O., Sibanda, T. C., Luo, Y. & Szovits, P. 2008. A de-identifier for medical discharge summaries. Artificial Intelligence in Medicine, 42**:**13-35.

[10] Gardner, J., Xiong, L., Wang, F., Post, A., Saltz, J. & Grandison, T, 2010. An evaluation of feature sets and sampling techniques for de-identification of medical records. In: Veinot T, (ed.), Proceedings of the 1st ACM International Health Informatics Symposium. New York:ACM. 183-190.

[11] Sweeney, L. 1996. Replacing personally-identifying information in medical records, the Scrub system. Proc AMIA Annu Fall Symp, 333-337.

[12] Taira, R. K., Bui, A. T. A. & Kangarloo, H. 2002. Identification of patient name references within medical documents using semantic selectional restrictions. Proc AMIA Annu Symp, 757-761.

[13] Ruch, P., Baud, R. H., Rassinoux, A. M., Bouillon, P. & Robert, G. 2000. Medical document anonymization with a semantic lexicon. Proc AMIA Symp, 729-733.

[14] Ferrández, Ó., South, B. R., Shen, S., Friedlin, F. J., Samore, M. H. & Meystre, S. M. 2013. BoB, a best-of-breed automated text de-identification system for VHA clinical documents. Journal of the American Medical Informatics Association, 20**:**77-83.

[15] Gupta, D., M. Saul, and J. Gilbertson. 2004. Evaluation of a deidentification (De-Id) software engine to share pathology reports and clinical documents for research. American Journal of Clinical Pathology 121 (2):176-186.

[16] Aramaki, E., Imai, T., Miyo, K., Ohe, K. Automatic deidentification by using sentence features and label consistency. In: Workshop on challenges in natural language I2b2 processing for clinical data. Washington, DC; 2006.

[17] Beckwith, B. A., R. Mahaadevan, U. J. Balis, and F. Kuo. 2006. Development and evaluation of an open source software tool for deidentification of pathology reports. BMC Medical Informatics and Decision Making 6.

[18] Guo, Y., Gaizauskas, R., Roberts, I., Demetriou, G., Hepple, M. 2006. Identifying personal health information using support vector machines. In: Workshop on challenges in natural language I2b2 processing for clinical data. Washington, DC; 2006.

[19] Friedlin, F. Jeff, and Clement J. McDonald. 2008. A software tool for removing patient identifying information from clinical documents. Journal of the American Medical Informatics Association 15 (5):601-610.

[20] Berman, J. J. 2003. Concept-match medical data scrubbing - How pathology text can be used in research. Archives of Pathology & Laboratory Medicine, 127**:**680-686.

[21] Morrison, F. P., Lai, A. M. & Hripcsak, G. 2009. Repurposing the Clinical Record: Can an Existing Natural Language Processing System De-identify Clinical Notes? Journal of the American Medical Informatics Association, 16**:**37-39.

[22] Meystre, S. M., Ferrández, Ó., Friedlin, F. J., South, B. R., Shen, S. & Samore, M. H. 2014. Text de-identification for privacy protection: A study of its impact on clinical text information content. Journal of Biomedical Informatics. doi: 10.1016/j.jbi.2014.01.011.

[23] Aberdeen, J., Bayer, S., Yeniterzi, R., Wellner, B., Clark, C., Hanauer, D., Malin, B. & Hirschman, L. 2010. The MITRE Identification Scrubber Toolkit: Design, training, and assessment. International Journal of Medical Informatics, 79**:**849-859.

[24] Kushida, C. A., Nichols, D. A., Jadrnicek, R., Miller, R., Walsh, J. K., & Griffin, K. 2012. Strategies for de-identification and anonymization of electronic health record data for use in multicenter research studies. Medical care, 50(Suppl 1):S82-S101.

[25] Uzuner, O., Luo, Y. & Szolovits, P. 2007. Evaluating the state-of-the-art in automatic de-identification. Journal of the American Medical Informatics Association, 14:550-563.

[26] Benton, A., Hill, S., Ungar, L., Chung, A., Leonard, C., Freeman, C. & Holmes, J. H. 2011. A system for de-identifying medical message board text. BMC Bioinformatics, 12(Suppl 3):S2.

[27] Courtois, B., Silberztein, M. 1990. Dictionnaires électroniques du françai̇s. Larousse, Paris.

[28] Gross, M. 1989. The use of finite automata in the lexical representation of natural language. Lecture Notes in Computer Science, 377:34-50.

[29] Paumier, S. 2011. Unitex 3.0 User manual. http:// http://www-igm.univ-mlv.fr/~unitex/UnitexManual3.0.pdf.

[30] Krstev, C. Processing of Serbian – Automata, Texts and Electronic dictionaries. Faculty of Philology, University of Belgrade, Belgrade, 2008.

[31] Krstev, C., Obradović, I., Utvić, M. & Vitas, D. 2014. A system for named entity recognition based

on local grammars. Journal of Logic and Computation, 24:473-489.

[32] Maurel, D., Friburger, N., Antoine, J. Y., Eshkol-Taravella, I. & Nouvel, D. 2011. Transducer cascades surrounding the recognition of named entities. Cascades de transducteurs autour de la reconnaissance des entités nommées, 52:69-96.

[33] Makhoul, J., Kubala, F., Schwartz, R., & Weischedel, R. 1999. Performance measures for information extraction. In: Proceedings of DARPA broadcast news workshop. 249-252.

[34] El Emam, K., Dankar, F. K., Issa, R., Jonker, E., Amyot, D., Coco, E., Corriveau, J.P., Walker, M., Chowdhury, S., Vaillancourt, R., Roffey, T., & Bottomley, J. 2009. A globally optimal k-anonymity method for the de-identification of health data. Journal of the American Medical Informatics Association, 16:670-82.

Probabilistic 2D Point Interpolation and Extrapolation via Data Modeling

Dariusz Jacek Jakóbczak
Department of Electronics and Computer Science, Technical University of Koszalin,
Sniadeckich 2, 75-453 Koszalin, Poland
E-mail: dariusz.jakobczak@tu.koszalin.pl

Keywords: 2D data interpolation, Hurwitz-Radon matrices, MHR method, probabilistic modeling, curve extrapolation

Received: June 25, 2014

Mathematics and computer science are interested in methods of 2D curve interpolation and extrapolation using the set of key points (knots). A proposed method of Hurwitz- Radon Matrices (MHR) is such a method. This novel method is based on the family of Hurwitz-Radon (HR) matrices which possess columns composed of orthogonal vectors. Two-dimensional curve is interpolated via different functions as probability distribution functions: polynomial, sinus, cosine, tangent, cotangent, logarithm, exponent, arcsin, arccos, arctan, arcctg or power function, also inverse functions. It is shown how to build the orthogonal matrix operator and how to use it in a process of curve reconstruction.

Povzetek: Opisana je nova metoda 2D interpolacije in ekstrapolacije krivulj.

1 Introduction

Curve interpolation and extrapolation [1] represents one of the most important problems in mathematics: how to model the curve [2] via discrete set of two-dimensional points [3]? Also the matter of curve representation and parameterization is still opened in mathematics and computer sciences [4]. The author wants to approach a problem of curve modeling by characteristic points. Proposed method relies on functional modeling of curve points situated between the basic set of the nodes or outside the nodes. The functions that are used in calculations represent whole family of elementary functions with inverse functions: polynomials, trigonometric, cyclometric, logarithmic, exponential and power function. These functions are treated as probability distribution functions in the range [0,1]. Nowadays methods apply mainly polynomial functions, for example Bernstein polynomials in Bezier curves, splines and NURBS [5]. Numerical methods for data interpolation or extrapolation are based on polynomial or trigonometric functions, for example Lagrange, Newton, Aitken and Hermite methods. These methods have some weak sides [6] and are not sufficient for curve interpolation and extrapolation in the situations when the curve cannot be build by polynomials or trigonometric functions. Proposed 2D curve interpolation and extrapolation is the functional modeling via any elementary functions and it helps us to fit the curve during the computations.

The main contributions of the paper are dealing with presentation the method that connects such problems as: interpolation, extrapolation, modeling, numerical methods and probabilistic methods. This is new approach to these problems. Differences from the previous papers of the author are connected with calculations without matrices ($N = 1$), new probabilistic distribution functions and novel look on shape modeling and curve reconstruction.

The method of Hurwitz-Radon Matrices (MHR) requires minimal assumptions: the only information about a curve is the set of at least two nodes. Proposed method of Hurwitz-Radon Matrices (MHR) is applied in curve modeling via different coefficients: polynomial, sinusoidal, cosinusoidal, tangent, cotangent, logarithmic, exponential, arcsin, arccos, arctan, arcctg or power. Function for MHR calculations is chosen individually at each interpolation and it represents probability distribution function of parameter $\alpha \in [0,1]$ for every point situated between two interpolation knots. MHR method uses two-dimensional vectors (x,y) for curve modeling - knots $(x_i,y_i) \in \boldsymbol{R}^2$ in MHR method:

1. MHR version with no matrices ($N = 1$) needs 2 knots or more;
2. At least five knots (x_1,y_1), (x_2,y_2), (x_3,y_3), (x_4,y_4) and (x_5,y_5) if MHR method is implemented with matrices of dimension $N = 2$;
3. For more precise modeling knots ought to be settled at key points of the curve, for example local minimum or maximum and at least one node between two successive local extrema.

Condition 2 is connected with important features of MHR method: MHR version with matrices of dimension $N = 2$ (MHR-2) requires at least five nodes, MHR version with matrices of dimension $N = 4$ (MHR-4) needs at least nine nodes and MHR version with matrices of dimension N= 8 (MHR-8) requires at least 17 nodes. Condition 3 means for example the highest point of the curve in a particular orientation, convexity changing or curvature extrema. So this paper wants to answer the question: how to interpolate end extrapolate the curve by a set of knots?

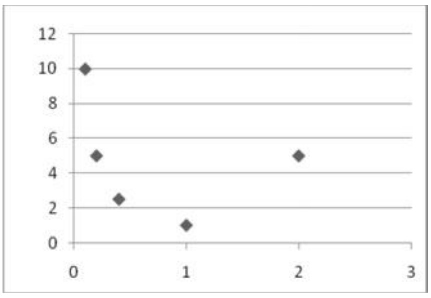

Figure 1: Knots of the curve before modelling.

Coefficients for curve modeling are computed using probability distribution functions: polynomials, power functions, sinus, cosine, tangent, cotangent, logarithm, exponent or arcsin, arccos, arctan, arcctg.

2 Probabilistic 2D interpolation and extrapolation

The method of Hurwitz – Radon Matrices (MHR) is computing points between two successive nodes of the curve: calculated points are interpolated and parameterized for real number $\alpha \in [0,1]$ in the range of two successive nodes. Data extrapolation is possible for $\alpha < 0$ or $\alpha > 1$. MHR calculations are dealing with square matrices of dimension N = 1, 2, 4 or 8. Matrices A_i, i=1,2…m satisfying

$$A_jA_k + A_kA_j = 0, \qquad A_j^2 = -I \qquad \text{for } j \neq k;\ j, k = 1,2...m$$

are called *a family of Hurwitz - Radon matrices*. They were discussed by Adolf Hurwitz and Johann Radon separately in 1923. A family of Hurwitz-Radon (HR) matrices [7] are skew-symmetric: $A_i^T = -A_i$ and $A_i^{-1} = -A_i$. Only for dimensions N = 1, 2, 4 or 8 the family of HR matrices consists of N - 1 matrices. For N = 1 there is no matrices but only calculations with real numbers. For N=2:

$$A_1 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

For N = 4 there are three HR matrices with integer entries:

$$A_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix}, \quad A_3 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix}.$$

For N = 8 we have seven HR matrices with elements 0, ±1. So far HR matrices have found applications in Space-Time Block Coding (STBC) [8] and orthogonal design [9], in signal processing [10] and Hamiltonian Neural Nets [11].

How coordinates of knots are applied for interpolation and extrapolation? If knots are represented by the following points $\{(x_i,y_i),\ i = 1, 2, \ldots, n\}$ then HR matrices combined with the identity matrix I_N are used to build the orthogonal Hurwitz - Radon Operator (OHR). For point $p_1=(x_1,y_1)$ and $x_1 \neq 0$ OHR of dimension N = 1 is the matrix (real number) M_1:

$$M_1(p_1) = \frac{1}{x_1^2}[x_1 \cdot y_1] = \frac{y_1}{x_1}. \qquad \textbf{(0)}$$

For points $p_1=(x_1,y_1)$ and $p_2=(x_2,y_2)$ OHR of dimension N=2 is build via matrix M_2:

$$M_2(p_1, p_2) = \frac{1}{x_1^2 + x_2^2}\begin{bmatrix} x_1y_1 + x_2y_2 & x_2y_1 - x_1y_2 \\ x_1y_2 - x_2y_1 & x_1y_1 + x_2y_2 \end{bmatrix}. \qquad \textbf{(1)}$$

For points $p_1=(x_1,y_1)$, $p_2=(x_2,y_2)$, $p_3=(x_3,y_3)$ and $p_4=(x_4,y_4)$ OHR M_4 of dimension N = 4 is introduced:

$$M_4(p_1, p_2, p_3, p_4) = \frac{1}{x_1^2 + x_2^2 + x_3^2 + x_4^2}\begin{bmatrix} u_0 & u_1 & u_2 & u_3 \\ -u_1 & u_0 & -u_3 & u_2 \\ -u_2 & u_3 & u_0 & -u_1 \\ -u_3 & -u_2 & u_1 & u_0 \end{bmatrix} \qquad \textbf{(2)}$$

where

$$u_0 = x_1y_1 + x_2y_2 + x_3y_3 + x_4y_4,$$
$$u_1 = -x_1y_2 + x_2y_1 + x_3y_4 - x_4y_3,$$
$$u_2 = -x_1y_3 - x_2y_4 + x_3y_1 + x_4y_2,$$
$$u_3 = -x_1y_4 + x_2y_3 - x_3y_2 + x_4y_1.$$

For knots $p_1=(x_1,y_1)$, $p_2=(x_2,y_2)$,… and $p_8=(x_8,y_8)$ OHR M_8 of dimension N = 8 is constructed [12] similarly as (1) and (2):

$$M_8(p_1, p_2...p_8) = \frac{1}{\sum_{i=1}^{8} x_i^2}\begin{bmatrix} u_0 & u_1 & u_2 & u_3 & u_4 & u_5 & u_6 & u_7 \\ -u_1 & u_0 & u_3 & -u_2 & u_5 & -u_4 & -u_7 & u_6 \\ -u_2 & -u_3 & u_0 & u_1 & u_6 & u_7 & -u_4 & -u_5 \\ -u_3 & u_2 & -u_1 & u_0 & u_7 & -u_6 & u_5 & -u_4 \\ -u_4 & -u_5 & -u_6 & -u_7 & u_0 & u_1 & u_2 & u_3 \\ -u_5 & u_4 & -u_7 & u_6 & -u_1 & u_0 & -u_3 & u_2 \\ -u_6 & u_7 & u_4 & -u_5 & -u_2 & u_3 & u_0 & -u_1 \\ -u_7 & -u_6 & u_5 & u_4 & -u_3 & -u_2 & u_1 & u_0 \end{bmatrix} \qquad \textbf{(3)}$$

where

$$\underline{u} = \begin{bmatrix} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 \\ -y_2 & y_1 & -y_4 & y_3 & -y_6 & y_5 & y_8 & -y_7 \\ -y_3 & y_4 & y_1 & -y_2 & -y_7 & -y_8 & y_5 & y_6 \\ -y_4 & -y_3 & y_2 & y_1 & -y_8 & y_7 & -y_6 & y_5 \\ -y_5 & y_6 & y_7 & y_8 & y_1 & -y_2 & -y_3 & -y_4 \\ -y_6 & -y_5 & y_8 & -y_7 & y_2 & y_1 & y_4 & -y_3 \\ -y_7 & -y_8 & -y_5 & y_6 & y_3 & -y_4 & y_1 & y_2 \\ -y_8 & y_7 & -y_6 & -y_5 & y_4 & y_3 & -y_2 & y_1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{bmatrix} \qquad \textbf{(4)}$$

and $\underline{u} = (u_0, u_1, \ldots, u_7)^T$ (4). OHR operators M_N (0)-(3) satisfy the condition of interpolation

$$M_N \cdot \mathbf{x} = \mathbf{y} \qquad \textbf{(5)}$$

for $\mathbf{x} = (x_1,x_2\ldots,x_N)^T \in \boldsymbol{R}^N$, $\mathbf{x} \neq \mathbf{0}$, $\mathbf{y} = (y_1,y_2\ldots,y_N)^T \in \boldsymbol{R}^N$ and N = 1, 2, 4 or 8.

2.1 Distribution functions in MHR interpolation and extrapolation

Points settled between the nodes are computed [13] using MHR method [14]. Each real number $c \in [a;b]$ is calculated by a convex combination

$$c = \alpha \cdot a + +(1 - \alpha) \cdot b \qquad \text{for}$$
$$\sim\ = \frac{b-c}{b-a} \in [0,1]. \qquad \textbf{(6)}$$

The weighted average OHR operator M of dimension N = 1, 2, 4 or 8 is build:

$$M = ¥ \cdot A + (1-¥) \cdot B \cdot \qquad \textbf{(7)}$$

The OHR matrix A is constructed (1)-(3) by every second knot $p_1=(x_1=a,y_1)$, $p_3=(x_3,y_3)$, …and $p_{2N\text{-}1}=(x_{2N\text{-}1},y_{2N\text{-}1})$:

$$A = M_N(p_1, p_{3,\ldots,}\, p_{2N\text{-}1}).$$

The OHR matrix B is computed (1)-(3) by knots $p_2=(x_2=b,y_2)$, $p_4=(x_4,y_4)$,… and $p_{2N}=(x_{2N},y_{2N})$:

$$B = M_N(p_2, p_{4,\ldots,}\, p_{2N}).$$

Vector of first coordinates C is defined for

$$c_i = \alpha \cdot x_{2i\text{-}1} + (1\text{-}\alpha) \cdot x_{2i} \quad , \quad i = 1, 2,\ldots, N \qquad \textbf{(8)}$$

and $C = [c_1, c_2,\ldots, c_N]^T$. The formula to calculate second coordinates $y(c_i)$ is similar to the interpolation formula (5):

$$Y(C) = M \cdot C \qquad \textbf{(9)}$$

where $Y(C) = [y(c_1), y(c_2),\ldots, y(c_N)]^T$. So interpolated value $y(c_i)$ from (9) depends on two, four, eight or sixteen ($2N$) successive nodes. For example N=1 results in computations without matrices:

$$A = M_1(p_1) = \frac{y_1}{x_1}, \quad B = M_1(p_2) = \frac{y_2}{x_2},$$
$$C = c_1 = \alpha \cdot x_1 + (1\text{-}\alpha) \cdot x_2 ,$$
$$Y(C) = y(c_1) = (¥ \frac{y_1}{x_1} + (1-¥)\frac{y_2}{x_2}) \cdot c_1,$$

$$y(c_1) = \sim \cdot ¥ \cdot y_1 + (1-\sim)(1-¥)y_2 + ¥(1-\sim)\frac{y_1}{x_1}x_2 + \sim(1-¥)\frac{y_2}{x_2}x_1 \quad . \qquad \textbf{(10)}$$

Formula (10) shows a clear calculation for interpolation of any point between two successive nodes (x_1,y_1) and (x_2,y_2). Key question is dealing with coefficient γ in (7). Basic MHR version means γ= α and then (10):

$$y(c_1) = \sim^2 \cdot y_1 + (1-\sim)^2 y_2 + \sim(1-\sim)(\frac{y_1}{x_1}x_2 + \frac{y_2}{x_2}x_1). \qquad \textbf{(11)}$$

Formula (11) represents the simplest way of MHR calculations (N = 1, γ = α) and it differs from linear interpolation $y(c) = \sim \cdot y_1 + (1-\sim)y_2$. MHR is not a linear interpolation.

Each interpolation requires specific distribution of parameter α (7) and γ depends on parameter α ∈ [0,1]:

$$\gamma = F(\alpha),\ F:[0,1] \rightarrow [0,1],\ F(0) = 0,\ F(1) = 1$$

and F is strictly monotonic.

Coefficient γ is calculated using different functions (polynomials, power functions, sinus, cosine, tangent, cotangent, logarithm, exponent, arcsin, arccos, arctan or arcctg, also inverse functions) and choice of function is connected with initial requirements and curve specifications. Different values of coefficient γ are connected with applied functions F(α). These functions (12)-(41) represent the probability distribution functions for random variable α ∈ [0,1] and real number $s > 0$:

1. power function

$$\gamma = \alpha^s \quad \text{with} \quad s > 0. \qquad \textbf{(12)}$$

For s = 1: basic version of MHR method when γ = α.

2. sinus

$$\gamma = sin(\alpha^s \cdot \pi/2)\ ,\ s > 0 \qquad \textbf{(13)}$$
or
$$\gamma = sin^s(\alpha \cdot \pi/2)\ ,\ s > 0. \qquad \textbf{(14)}$$

For s = 1: $\gamma = sin(\alpha \cdot \pi/2)$. **(15)**

3. cosine

$$\gamma = 1\text{-}cos(\alpha^s \cdot \pi/2)\ ,\ s > 0 \qquad \textbf{(16)}$$
or
$$\gamma = 1\text{-}cos^s(\alpha \cdot \pi/2)\ ,\ s > 0. \qquad \textbf{(17)}$$

For s = 1: $\gamma = 1\text{-}cos(\alpha \cdot \pi/2)$. **(18)**

4. tangent

$$\gamma = tan(\alpha^s \cdot \pi/4)\ ,\ s > 0 \qquad \textbf{(19)}$$
or
$$\gamma = tan^s(\alpha \cdot \pi/4)\ ,\ s > 0. \qquad \textbf{(20)}$$

For s = 1: $\gamma = tan(\alpha \cdot \pi/4)$. **(21)**

5. logarithm

$$\gamma = log_2(\alpha^s + 1)\ ,\ s > 0 \qquad \textbf{(22)}$$
or
$$\gamma = log_2{}^s(\alpha + 1)\ ,\ s > 0. \qquad \textbf{(23)}$$

For s = 1: $\gamma = log_2(\alpha + 1)$. **(24)**

6. exponent

$$¥ = (\frac{a^{\sim} - 1}{a-1})^s \ ,\ s > 0 \text{ and } a > 0 \text{ and } a \neq 1. \qquad \textbf{(25)}$$

For s = 1 and a = 2: $\gamma = 2^\alpha - 1$. **(26)**

7. arc sine

$$\gamma = 2/\pi \cdot arcsin(\alpha^s)\ ,\ s > 0 \qquad \textbf{(27)}$$
or
$$\gamma = (2/\pi \cdot arcsin\ \alpha)^s\ ,\ s > 0. \qquad \textbf{(28)}$$

For s = 1: $\gamma = 2/\pi \cdot arcsin(\alpha)$. **(29)**

8. arc cosine

$$\gamma = 1\text{-}2/\pi \cdot arccos(\alpha^s)\ ,\ s > 0 \qquad \textbf{(30)}$$
or
$$\gamma = 1\text{-}(2/\pi \cdot arccos\ \alpha)^s\ ,\ s > 0. \qquad \textbf{(31)}$$

For s = 1: $\gamma = 1\text{-}2/\pi \cdot arccos(\alpha)$. **(32)**

9. arc tangent

$$\gamma = 4/\pi \cdot arctan(\alpha^s)\ ,\ s > 0 \qquad \textbf{(33)}$$
or
$$\gamma = (4/\pi \cdot arctan\ \alpha)^s\ ,\ s > 0. \qquad \textbf{(34)}$$

For s = 1: $\gamma = 4/\pi \cdot arctan(\alpha)$. **(35)**

10. cotangent

$$\gamma = ctg(\pi/2 - \alpha^s \cdot \pi/4)\ ,\ s > 0 \qquad \textbf{(36)}$$
or
$$\gamma = ctg^s\ (\pi/2 - \alpha \cdot \pi/4),\ s > 0. \qquad \textbf{(37)}$$

For s = 1: $\gamma = ctg(\pi/2 - \alpha \cdot \pi/4)$. **(38)**

11. arc cotangent

$$\gamma = 2 - 4/\pi\cdot \mathit{arcctg}(\alpha^s)\ ,\ \ s > 0 \qquad \textbf{(39)}$$

or

$$\gamma = (2 - 4/\pi\cdot \mathit{arcctg}\ \alpha)^s\ ,\ \ s > 0. \qquad \textbf{(40)}$$

For $s = 1$: $$\gamma = 2 - 4/\pi\cdot \mathit{arcctg}(\alpha). \qquad \textbf{(41)}$$

Functions used in γ calculations (12)-(41) are strictly monotonic for random variable $\alpha \in [0,1]$ as $\gamma = F(\alpha)$ is probability distribution function. Also inverse function $F^{-1}(\alpha)$ is appropriate for γ calculations. Choice of function and value *s* depends on curve specifications and individual requirements. Interpolating of coordinates for curve points using (6)-(9) is called by author the method of Hurwitz - Radon Matrices (MHR) [15]. So here are five steps of MHR interpolation:

Step 1: Choice of knots at key points.

Step 2: Fixing the dimension of matrices *N* = 1, 2, 4 or 8: *N* = 1 is the most universal for calculations (it needs only two nodes to compute unknown points between them) and it has the lowest computational costs (10); MHR with *N* = 2 uses four successive nodes to compute unknown coordinate; MHR version for *N* = 4 applies eight successive nodes to get unknown point and MHR with *N* = 8 needs sixteen successive nodes to calculate unknown coordinate (it has the biggest computational costs).

Step 3: Choice of distribution γ = F(α): basic distribution for γ = α.

Step 4: Determining values of α: α = 0.1, 0.2…0.9 (nine points) or 0.01, 0.02…0.99 (99 points) or others.

Step 5: The computations (9).
These five steps can be treated as the algorithm of MHR method of curve modeling and interpolation (6)-(9).

Considering nowadays used probability distribution functions for random variable α∈[0,1] - one distribution is dealing with the range [0,1]: beta distribution. Probability density function *f* for random variable α ∈ [0,1] is:

$$f(\sim) = c\cdot\sim^{s}\cdot(1-\sim)^{r}\ \ , s \ge 0, r \ge 0 \qquad \textbf{(42)}$$

When *r* = 0 probability density function (42) represents $f(\sim) = c\cdot\sim^{s}$ and then probability distribution function *F* is like (12), for example $f(\sim) = 3\sim^{2}$ and $\gamma=\alpha^3$. If *s* and *r* are positive integer numbers then γ is the polynomial, for example $f(\sim) = 6\sim(1-\sim)$ and $\gamma = 3\alpha^2\text{-}2\alpha^3$. So beta distribution gives us coefficient γ in (7) as polynomial because of interdependence between probability density *f* and distribution *F* functions:

$$f(\sim) = F'(\sim)\ \ ,\ F(\sim) = \int_0^{\sim} f(t)dt\,\cdot \qquad \textbf{(43)}$$

For example (43): $f(\sim) = \sim\cdot e^{\sim}$ and $¥ = F(\sim) = (\sim -1)e^{\sim} + 1\cdot$

What is very important: two curves may have the same set of nodes but different *N* or γ results in different interpolations (Fig.6-13). Here are some applications of MHR method with basic version (γ = α): MHR-2 is MHR version with matrices of dimension *N* = 2 and MHR-4 means MHR version with matrices of dimension *N* = 4.

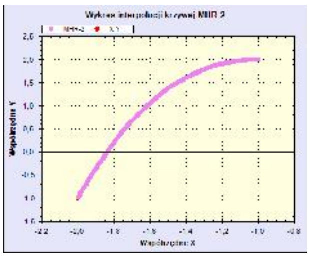

Figure 2: Function $f(x) = x^3+x^2-x+1$ with 396 interpolated points using basic MHR-2 with 5 nodes.

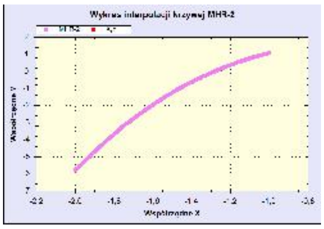

Figure 3: Function $f(x) = x^3+\ln(7\text{-}x)$ with 396 interpolated points using basic MHR-2 with 5 nodes.

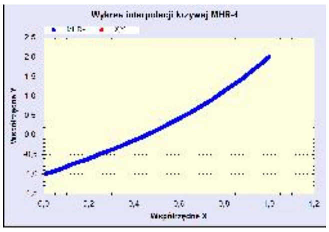

Figure 4: Function $f(x) = x^3+2x\text{-}1$ with 792 interpolated points using basic MHR-4 with 9 nodes.

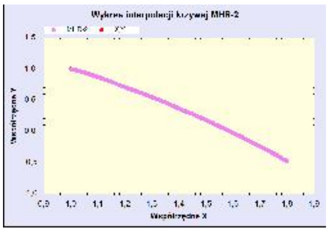

Figure 5: Function $f(x) = 3\text{-}2^x$ with 396 interpolated points using basic MHR-2 with 5 nodes.

Figures 2-5 show interpolation of continues functions connected with determined formula. So these functions are interpolated and modeled. Without knowledge about the formula, curve interpolation has to implement the coefficients γ (12)-(43), but MHR is not limited only to these coefficients. Each strictly monotonic function *F* between points (0;0) and (1;1) can be used in MHR interpolation. MHR 2D data extrapolation is possible for α < 0 or α>1.

3 Implementations of 2D probabilistic interpolation

Curve knots (0.1;10), (0.2;5), (0.4;2.5), (1;1) and (2;5) from Fig.1 are used in some examples of MHR interpolation with different γ. Points of the curve are calculated for *N* = 1 and γ = α (11) in example 1 and with matrices of dimension $N = 2$ in examples 2 - 8 for $\alpha = 0.1, 0.2, \ldots, 0.9$.

Example 1
Curve interpolation for $N = 1$ and $\gamma = \alpha$.

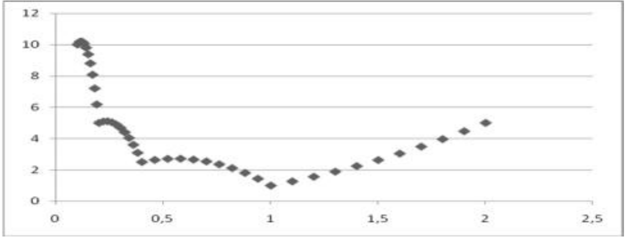

Figure 6: Modeling without matrices ($N = 1$) for nine reconstructed points between nodes.

Example 2
Sinusoidal interpolation with $\gamma = sin(\alpha \cdot \pi/2)$.

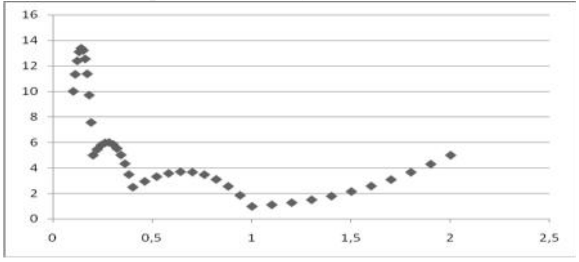

Figure 7: Sinusoidal modeling with nine reconstructed curve points between nodes.

Example 3
Tangent interpolation for $\gamma = tan(\alpha \cdot \pi/4)$.

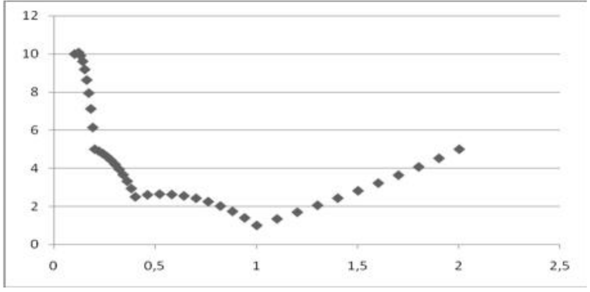

Figure 8: Tangent curve modeling with nine interpolated points between nodes.

Example 4
Tangent interpolation with $\gamma = tan(\alpha^s \cdot \pi/4)$ and $s = 1.5$.

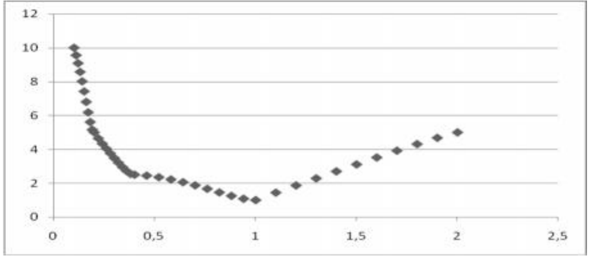

Figure 9: Tangent modeling with nine recovered points between nodes.

Example 5
Tangent curve interpolation for $\gamma = tan(\alpha^s \cdot \pi/4)$ and $s = 1.797$.

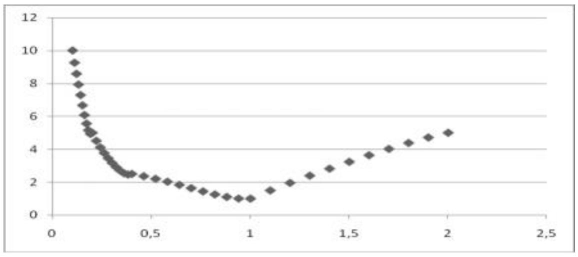

Figure 10: Tangent modeling with nine reconstructed points between nodes.

Example 6
Sinusoidal interpolation with $\gamma = sin(\alpha^s \cdot \pi/2)$ and $s = 2.759$.

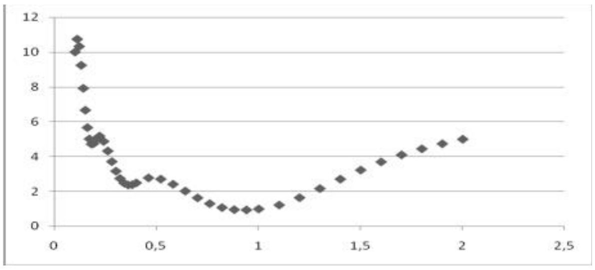

Figure 11: Sinusoidal modeling with nine interpolated curve points between nodes.

Example 7
Power function modeling for $\gamma = \alpha^s$ and $s = 2.1205$.

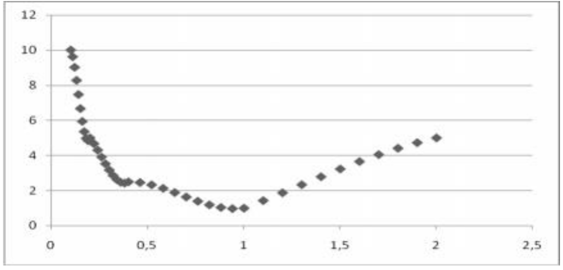

Figure 12: Power function curve modeling with nine recovered points between nodes.

Example 8
Logarithmic curve modeling with $\gamma = log_2(\alpha^s + 1)$ and $s = 2.533$.

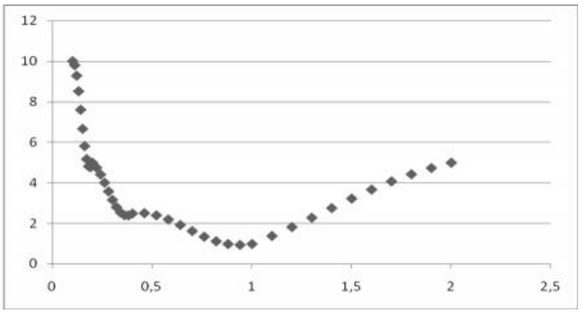

Figure 13: Logarithmic modeling with nine reconstructed points between nodes.

These eight examples demonstrate possibilities of curve interpolation for key nodes. Reconstructed values and interpolated points, calculated by MHR method, are applied in the process of curve modeling. Every curve can be interpolated by some distribution function as parameter γ. This parameter is treated as probability distribution function for each curve.

4 MHR 2D modeling versus polynomial interpolation

4.1 Example 4.1

Let us consider a graph of function *f*(*x*) = 2/*x* in range [0.4,1.6]. There are given five interpolation nodes for *x* = 0.4, 0.7, 1.0, 1.3, 1.6. The curve *y* = 2/*x* reconstructed by basic MHR method (12) with *N* = 2 for *s* = 1 looks not precisely:

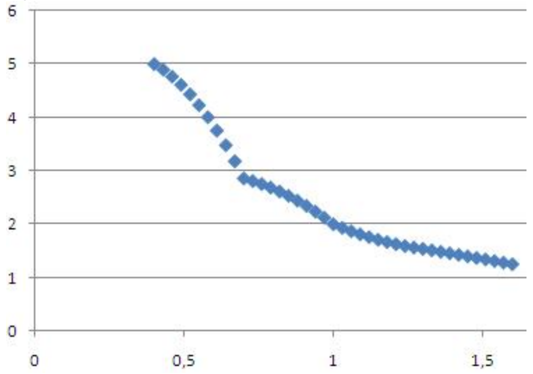

Figure 14: The curve *y* = 2/*x* reconstructed by MHR method for γ = α (7) and five nodes together with 36 computed points.

Lagrange interpolation polynomial is not to be accepted:

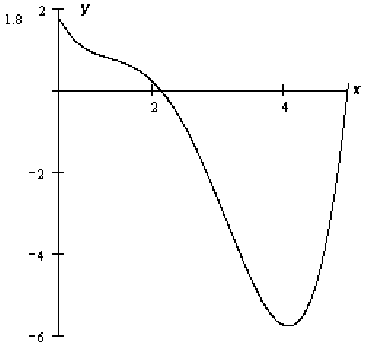

Figure 15: Polynomial interpolation of function *y* = 1/*x* is out of acceptation.

For better reconstruction of the curve, appropriate parameter *s* in MHR method (12) is calculated. Choice of parameter *s* is connected with comparison of accurate values w_i for function *f*(*x*) = 2/*x* in control points p_i, situated in the middle between interpolation nodes (α=0.5), and values in control points p_i computed by MHR method. Control points are settled in the middle between interpolation nodes, because interpolation error of MHR method is the biggest. Choice of coefficient *s* is done by criterion: difference between precise values w_i and values reconstructed by MHR method is the smallest. Control points p_i in this example are established for x_i = 0.55, 0.85, 1.15, 1.45. Four values of the curve are compared for parameter *s* = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0. The smallest difference is calculated for *s* = 1.5:

$$|w1 - 3.694| + |w3 - 1.632| + |w2 - 2.302| + |w4 - 1.329| = 0.266 \quad \textbf{(44)}$$

Calculations for average OHR operator *M* (7) are done:

$$M = \begin{bmatrix} 2.405 & 2.062 \\ -2.062 & 2.405 \end{bmatrix}, \quad M \cdot \begin{bmatrix} p1 \\ p3 \end{bmatrix} = \begin{bmatrix} 3.694 \\ 1.632 \end{bmatrix};$$

$$M = \begin{bmatrix} 1.375 & 0.782 \\ -0.782 & 1.375 \end{bmatrix}, \quad M \cdot \begin{bmatrix} p2 \\ p4 \end{bmatrix} = \begin{bmatrix} 2.302 \\ 1.329 \end{bmatrix}.$$

Computed values appear in (44). Other results:

a) *s*=0.1

$$|w1 - 5.815| + |w3 - 1.939| + |w2 - 3.209| + |w4 - 1.601| = 3.456$$

b) *s*=0.2:

$$|w1 - 5.587| + |w3 - 1.906| + |w2 - 3.111| + |w4 - 1.572| = 3.068$$

c) *s*=0.3:

$$|w1 - 5.373| + |w3 - 1.875| + |w2 - 3.02| + |w4 - 1.544| = 2.704$$

d) *s*=0.4:

$$|w1 - 5.174| + |w3 - 1.846| + |w2 - 2.935| + |w4 - 1.519| = 2.366$$

e) *s*=0.5:

$$|w1 - 4.988| + |w3 - 1.819| + |w2 - 2.856| + |w4 - 1.495| = 2.05$$

f) *s*=0.6:

$$|w1 - 4.815| + |w3 - 1.794| + |w2 - 2.782| + |w4 - 1.473| = 1.756$$

g) *s*=0.7:

$$|w1 - 4.653| + |w3 - 1.771| + |w2 - 2.712| + |w4 - 1.452| = 1.48$$

h) *s*=0.8:

$$|w1 - 4.503| + |w3 - 1.749| + |w2 - 2.648| + |w4 - 1.433| = 1.225$$

i) *s*=0.9:

$$|w1 - 4.362| + |w3 - 1.728| + |w2 - 2.588| + |w4 - 1.415| = 1.008$$

j) *s*=1:

$$|w1 - 4.23| + |w3 - 1.709| + |w2 - 2.532| + |w4 - 1.398| = 0.822$$

k) *s*=1.1:

$$|w1 - 4.108| + |w3 - 1.692| + |w2 - 2.479| + |w4 - 1.382| = 0.648$$

l) *s*=1.2:

$$|w1 - 3.994| + |w3 - 1.675| + |w2 - 2.43| + |w4 - 1.368| = 0.51$$

m) *s*=1.3:

$$|w1 - 3.887| + |w3 - 1.66| + |w2 - 2.385| + |w4 - 1.354| = 0.387$$

n) *s*=1.4:

$$|w1 - 3.787| + |w3 - 1.645| + |w2 - 2.342| + |w4 - 1.341| = 0.294$$

o) *s*=1.6:

$$|w1 - 3.608| + |w3 - 1.619| + |w2 - 2.265| + |w4 - 1.318| = 0.298$$

p) *s*=1.7:

$$|w1 - 3.527| + |w3 - 1.608| + |w2 - 2.231| + |w4 - 1.308| = 0.434$$

q) *s*=1.8:

$$|w1 - 3.451| + |w3 - 1.597| + |w2 - 2.199| + |w4 - 1.298| = 0.563$$

r) *s*=1.9:

$$|w1 - 3.381| + |w3 - 1.587| + |w2 - 2.169| + |w4 - 1.289| = 0.682$$

s) *s*=2.0:

$$|w1 - 3.315| + |w3 - 1.577| + |w2 - 2.14| + |w4 - 1.281| = 0.795$$

One can see the changes of interpolation error. Reconstruction of the curve *y*=2/*x* by MHR method for N = 2 and γ = $\alpha^{1.5}$ looks as follows:

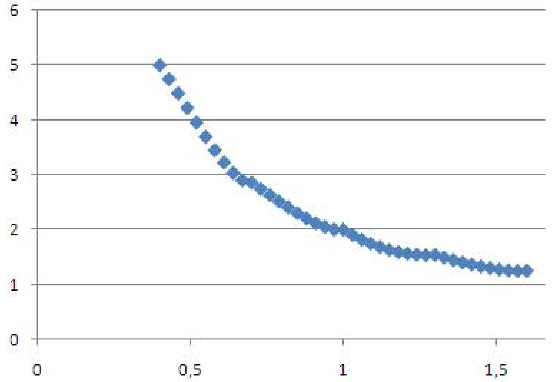

Figure 16: The curve $y = 2/x$ modeled via MHR method (12) for s = 1.5 and five nodes together with 36 reconstructed points.

Figure 16 represents the curve $y = 2/x$ more precisely then classical interpolation and Figure 14. If we would like to have better parameter s (with two digits after coma), calculations are done:

a) s=1.49:

$$|w1-3.703|+|w3-1.633|+|w2-2.306|+|w4-1.33|=0.269$$

b) s=1.51:

$$|w1-3.685|+|w3-1.631|+|w2-2.299|+|w4-1.328|=0.262$$

c) s=1.52:

$$|w1-3.677|+|w3-1.629|+|w2-2.295|+|w4-1.327|=0.261$$

d) s=1.53:

$$|w1-3.668|+|w3-1.628|+|w2-2.291|+|w4-1.326|=0.258$$

e) s=1.54:

$$|w1-3.659|+|w3-1.627|+|w2-2.287|+|w4-1.325|=0.255$$

f) s=1.55:

$$|w1-3.65|+|w3-1.626|+|w2-2.284|+|w4-1.324|=0.251$$

g) s=1.56:

$$|w1-3.642|+|w3-1.624|+|w2-2.28|+|w4-1.323|=0.25$$

h) s=1.57:

$$|w1-3.633|+|w3-1.623|+|w2-2.276|+|w4-1.321|=0.255$$

i) s=1.58:

$$|w1-3.625|+|w3-1.622|+|w2-2.273|+|w4-1.32|=0.268$$

j) s=1.59:

$$|w1-3.616|+|w3-1.621|+|w2-2.269|+|w4-1.319|=0.283$$

Model of the curve for s = 1.56 in (12) is more accurate then with s = 1.5.

Convexity of reconstructed curve is very important factor in MHR method with parameter s. Appropriate choice of parameter s is connected with regulation and controlling of convexity: model of the curve (Fig. 16) preserves monotonicity and convexity.

4.2 Example 4.2

This example considers a graph of function $f(x) = 1/(1+5x^2)$ in the range [-1,1]:

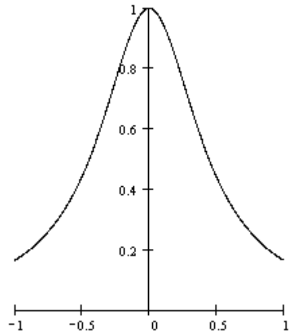

Figure 17: A graph of function $f(x) = 1/(1+5x^2)$ in range [-1,1].

There are given five interpolation nodes for x = -1.0, -0.5, 0, 0.5, 1.0. It is an example of function with useless Lagrange interpolation polynomial: Runge phenomenon and unpleasant two minima.

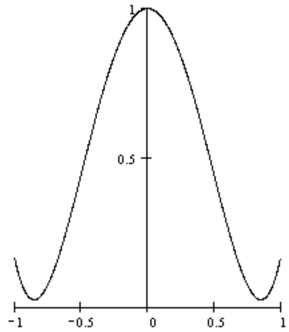

Figure 18: Lagrange interpolation polynomial differs extremely from a graph of function $f(x) = 1/(1+5x^2)$.

Model of the curve $y = 1/(1+5x^2)$ in basic MHR method (12) for N = 2 and s = 1:

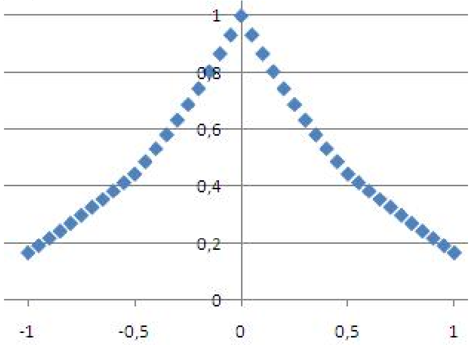

Figure 19: The curve $y = 1/(1+5x^2)$ modeled via MHR method for γ = α (7) and five nodes together with 36 reconstructed points.

Reconstructed curve (Fig. 19) preserves monotonicity and symmetry for s = 1. Comparing precise values w_i with values computed by MHR method in control points p_i, fixed for x_i = -0.75, -0.25, 0.25, 0.75, identical the best results appear for s = 1 (12):

$$|w1-0.299|+|w3-0.688|+|w2-0.688|+|w4-0.299|=0.221 \quad (45)$$

and for s = 0.5, 0.6, 0.7, 0.8, 0.9, 1.1, 1.2, 1.3, 1.4, 1.5, 1.7, 1.8, 1.9. But only for s = 1 reconstructed curve is symmetric- look at values in (45). So if a case s = 1 appears among the best results, then model ought to be built for s = 1. And if a case s = 1 does not appear among the best results, then model ought to be built for s near by 1. Here is an example of reconstructed curve for s=1.5 (without feature of symmetry):

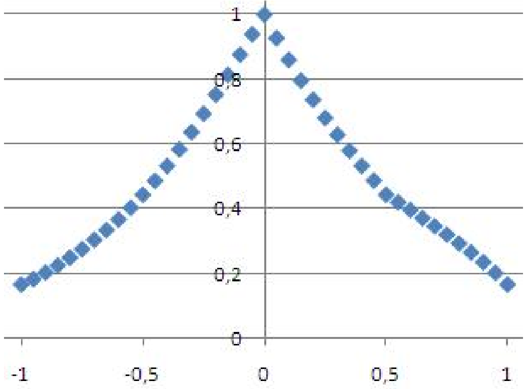

Figure 20: The curve $y = 1/(1+5x^2)$ modeled via MHR method for s = 1.5 (12) and five nodes together with 36 reconstructed points.

The best model of the curve $y = 1/(1+5x^2)$, built by MHR method for s = 1 (Fig. 19), preserves monotonicity and symmetry. Convexity of the function (Fig. 17) can make some troubles.

5 Future research directions

Future works with MHR method are connected with object recognition after geometrical transformations of curve (translations, rotations, scaling)- only nodes are transformed and new curve (for example contour of the object) for new nodes is reconstructed. Also estimation of object area in the plane, using nodes of object contour, will be possible by MHR interpolation. Object area is significant feature for object recognition. Future works are dealing with smoothing the curve, parameterization of whole curve and possibility to apply MHR method to three-dimensional curves. Also case of equidistance nodes must be studied with all details. Another future research direction is to apply MHR method in artificial intelligence and computer vision, for example identification of a specific person’s face or fingerprint, optical character recognition (OCR), image restoration, content-based image retrieval and pose estimation. Future works are connected with object recognition for any set of contour points. There are several specialized tasks based on recognition to consider and it is important to use the shape of whole contour for identification and detection of persons, vehicles or other objects. Other applications of MHR method will be directed to computer graphics, modeling and image processing.

6 Conclusion

The method of Hurwitz-Radon Matrices (MHR) enables interpolation of two-dimensional curves using different coefficients γ: polynomial, sinusoidal, cosinusoidal, tangent, cotangent, logarithmic, exponential, arcsin, arccos, arctan, arcctg or power function [16], also inverse functions. Function for γ calculations is chosen individually at each curve modeling and it is treated as probability distribution function: γ depends on initial requirements and curve specifications. MHR method leads to curve interpolation via discrete set of fixed knots. So MHR makes possible the combination of two important problems: interpolation and modeling. Main features of MHR method are:

a) the smaller distance between knots the better;
b) calculations for coordinate x close to zero and near by extremum require more attention;
c) MHR interpolation of the function is more precise then linear interpolation;
d) minimum two interpolation knots for calculations without matrices when N=1, but MHR is not a linear interpolation;
e) interpolation of L points is connected with the computational cost of rank $O(L)$;
f) MHR is well-conditioned method (orthogonal matrices)[17];
g) coefficient γ is crucial in the process of curve probabilistic interpolation and it is computed individually for a single curve;
h) MHR 2D data extrapolation is possible for $\alpha < 0$ or $\alpha > 1$.

Future works are going to: choice and features of coefficient γ, implementation of MHR in object recognition [18], shape geometry, contour modeling and parameterization [19].

References

[1] Collins II, G.W.: Fundamental Numerical Methods and Data Analysis. Case Western Reserve University (2003)

[2] Chapra, S.C.: Applied Numerical Methods. McGraw-Hill (2012)

[3] Ralston, A., Rabinowitz, P.: A First Course in Numerical Analysis – Second Edition. Dover Publications, New York (2001)

[4] Zhang, D., Lu, G.: Review of Shape Representation and Description Techniques. Pattern Recognition 1(37), 1-19 (2004)

[5] Schumaker, L.L.: Spline Functions: Basic Theory. Cambridge Mathematical Library (2007)

[6] Dahlquist, G., Bjoerck, A.: Numerical Methods. Prentice Hall, New York (1974)

[7] Eckmann, B.: Topology, Algebra, Analysis-Relations and Missing Links. Notices of the American Mathematical Society 5(46), 520-527 (1999)

[8] Citko, W., Jakóbczak, D., Sieńko, W.: On Hurwitz - Radon Matrices Based Signal Processing. Workshop Signal Processing at Poznan University of Technology (2005)

[9] Tarokh, V., Jafarkhani, H., Calderbank, R.: Space-Time Block Codes from Orthogonal Designs. IEEE Transactions on Information Theory 5(45), 1456-1467 (1999)

[10] Sieńko, W., Citko, W., Wilamowski, B.: Hamiltonian Neural Nets as a Universal Signal Processor. 28th Annual Conference of the IEEE Industrial Electronics Society IECON (2002)

[11] Sieńko, W., Citko, W.: Hamiltonian Neural Net Based Signal Processing. The International Conference on Signal and Electronic System ICSES (2002)

[12] Jakóbczak, D.: 2D and 3D Image Modeling Using Hurwitz-Radon Matrices. Polish Journal of Environmental Studies 4A(16), 104-107 (2007)

[13] Jakóbczak, D.: Shape Representation and Shape Coefficients via Method of Hurwitz-Radon Matrices. Lecture Notes in Computer Science 6374 (Computer Vision and Graphics: Proc. ICCVG 2010, Part I), Springer-Verlag Berlin Heidelberg, 411-419 (2010)

[14] Jakóbczak, D.: Curve Interpolation Using Hurwitz-Radon Matrices. Polish Journal of Environmental Studies 3B(18), 126-130 (2009)

[15] Jakóbczak, D.: Application of Hurwitz-Radon Matrices in Shape Representation. In: Banaszak, Z., Świć, A. (eds.) Applied Computer Science: Modelling of Production Processes 1(6), pp. 63-74. Lublin University of Technology Press, Lublin (2010)

[16] Jakóbczak, D.: Object Modeling Using Method of Hurwitz-Radon Matrices of Rank k. In: Wolski, W., Borawski, M. (eds.) Computer Graphics: Selected Issues, pp. 79-90. University of Szczecin Press, Szczecin (2010)

[17] Jakóbczak, D.: Implementation of Hurwitz-Radon Matrices in Shape Representation. In: Choraś, R.S. (ed.) Advances in Intelligent and Soft Computing 84, Image Processing and Communications: Challenges 2, pp. 39-50. Springer-Verlag, Berlin Heidelberg (2010)

[18] Jakóbczak, D.: Object Recognition via Contour Points Reconstruction Using Hurwitz-Radon Matrices. In: Józefczyk, J., Orski, D. (eds.) Knowledge-Based Intelligent System Advancements: Systemic and Cybernetic Approaches, pp. 87-107. IGI Global, Hershey PA, USA (2011)

[19] Jakóbczak, D.: Curve Parameterization and Curvature via Method of Hurwitz-Radon Matrices. Image Processing & Communications- An International Journal 1-2(16),49-56,(2011)

Swarm Intelligence and its Application in Abnormal Data Detection

Bo Liu, Mei Cai and Jiazong Yu
College of Information Science and Technology, Jinan University, Guangzhou, China
E-mail: ddxllb@163.com

Keywords: swarm intelligence, abnormal data, detection, ACO, PSO, BCO

Received: June 18, 2014

This study addresses swarm intelligence-based approaches in data quality detection. First, three typical swarm intelligence models and their applications in abnormity detection are introduced, including Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Bee Colony Optimization (BCO). Then, it presents three approaches based on ACO, PSO and BCO for detection of attribute outliers in datasets. These approaches use different search strategies on the data items; however, they choose the same fitness function (i.e. the O-measure) to evaluate the solutions, and they make use of swarms of the fittest agents and random moving agents to obtain superior solutions by changing the searching paths or positions of agents. Three algorithms are described and explained, which are efficient by heuristic principles.

Povzetek: Opisane so tri metode z roji za analizo kvalitet podatkov: na osnovi mravelj, delcev in čebel.

1 Introduction

Swarm Intelligence (SI) is a branch of Artificial Intelligence that is used to model the collective behaviour of social swarms (groups of agents) in nature, such as ant colonies, honey bees and bird flocks[1]. Each agent can not only interact with its local environment and other agents, but also respond to environmental change immediately, such that it can always find its objective. Swarm-based algorithms have emerged as a family of nature-inspired, population-based algorithms that are capable of producing low-cost, fast and robust solutions to several complex problems[2][3]. [4][5][6] introduce a slice of new developments made in the theory and applications of bio-inspired computation. For instance, Cui et al.[7] proposed the artificial plant optimization algorithm with detailed case studies, and Yang et al.[8] edited a special issue on meta-heuristics and swarm intelligence in engineering and industry.

Data quality mining (DQR) is a new and promising application of data mining techniques for the detection, quantification and correction of data quality deficiencies in very large databases. Typical data mining techniques, such as clustering, classification, association analysis, have been used to discover abnormal data in databases[9][10]. Recently swarm-based DQR approaches have also been developed. For example, Alam et al.[11] proposed SI-based clustering approaches for outlier detection, and Jindal et al.[12] surveyed of SI in intrusion detection.

The motivation of this study is dedicated to the application of SI in data quality mining, which explores new approaches for abnormal data discovery, avoiding searching the whole data space, such that improves efficiency and accuracy in abnormal data detection. The paper introduces typical SI models, and investigate their applications in abnormal data or behaviour detection. It discusses the inadequacy of current SI-based solutions in abnormal data detection, and proposes new approaches utilizing three SI models for detection of abnormal attributes.

The rest of this paper is organized as follows: in Section 2, typical SI models are introduced. We give a survey of SI-based methods for outlier detection in Section 3. New approaches based on SI for abnormal attribute detection are presented in Section 4, and the paper concludes with our final remarks in Section 5.

2 Typical SI models

SI was first introduced by G. Beni and J. Wang in the global optimization framework as a set of algorithms for controlling robotic swarm [13], which has become one of popular natural computing[14] areas. The most popular models of SI include: Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO) and Bee Colony Optimization (BCO). The main ideas of the three models are discussed below.

There is no single ‘Ant Model’, rather there is a family of models, each inspired by a different aspect of ant behaviour[15]. These models include those inspired by ant-foraging behaviour, brood-sorting behaviour, cemetery formation behaviour and cooperative transport. One typical behaviour is ant foraging. Each ant deposits a chemical substance (called a pheromone) on the ground while walking[16], and the pheromone encourages the following ants to stay close to the previous path. Meanwhile the pheromone evaporates over time, allowing search exploration. Dorigo and Maniezzo illustrated the complex behaviour of ant colonies through a number of experiments in [17], which showed that the ants following the shorter path will make more visits to the source than those following the longer path. ACO is inspired by observation of the behaviour of real ants, which was proposed by Dorigo and colleagues[18] as a method for solving combinatorial optimization problems.

PSO was introduced by J. Kennedy et al.[19]. This is a population-based optimization technique inspired by the social behaviour of bird flocking and fish schooling. It was observed that large numbers of birds often change direction suddenly, and scatter and regroup. The synchrony of flocking behaviour is thought to be a function of the birds' efforts to maintain an optimum distance between themselves and their neighbours. PSO researchers have implemented such natural processes to solve optimization problems in which each single solution, called a particle, joins other individuals to make up a swarm (population) for exploring within the search space. Each particle has a fitness value calculated by a fitness function, and a velocity of moving towards the optimum[20]. The PSO process starts by creating random particles and initializing the best positions for each particle and for the whole population. It then iterates the computation of the positions and velocities of each particle movement and the update of the local best position for each particle and the global best position for the entire swarm. PSO has been continually modified trying to improve its convergence properties, and the PSO variants have been used to solve a wide range of optimization and inverse problems [21].

BCO was proposed by D. Karabago as a new member of the family of SI algorithms[22]. In nature, bees reach a nectar source by following a nest-mate who has already discovered a patch of flowers. When a forager bee (recruiter) decides to attract more bee mates to a newly-discovered good food source, it returns to the hive and starts performing what is known as the waggle dance to communicate spatial and profitability information about the discovered food source, and recruit more honey bees (dancer followers) to exploit it [1]. BCO is based on a swarm of artificial bees cooperating together to solve a problem. First, a bee, named InitBee, settles to find a solution with good features. From this first solution a set of other solutions called SearchArea is determined by using a certain strategy. Then, every bee will consider a solution from SearchArea as its starting point in the search, and communicates the best visited solution to all its neighbours. However, if after a period of time the swarm observes that the solution is not improved, it introduces a criterion of diversification, preventing it from being trapped in a local optimum [23].

3 Survey of SI-based methods for outlier detection

An abnormity is an object that does not conform to the normal behaviour of a dataset. Because abnormal data are very much in the minority in a dataset, they are also called outliers or noises. Although abnormal data or outliers may not be errors, detecting them is the most valid way to eliminate errors. There are two kinds of outliers: class outliers and attribute outliers[24]. Class outliers are records or tuples belonging to rare categories or labelled with wrong classes, and attribute outliers are attribute values deviating from the normal distribution or error attribute values in dataset records. In other words, they are at two levels: the record level and the attribute value level.

Many studies have used traditional data mining or machine learning techniques to discover outliers, which need training on classifying rules, data patterns, or clustering instances of the dataset. There are also several approaches based on SI without training for detecting outliers, which are simple and heuristic. We focus on SI-based solutions in the rest of this study.

In [12], an ACO-based approach of clustering was proposed as a data preprocessing procedure, during which outliers can be detected. In this process, the continuity of ants for similar data points and dissimilar data points is collected into the next nodes. Each ant of the data points compares their property values according to the initial data point set, checks the importance of the data points and iteratively updates the values of the pheromones laid by other ants. Lastly, the data point selection matrix for obtaining final clustering results using another algorithm is generated; at the same time, those unselected data points are outliers. Without setting the number of clusters and initial centre points, the ACO-based clustering method can obtain the high clustering correctness rate and outlier filtering rate.

Alam et al.[11] proposed a SI-based clustering technique called Hierarchical PSO Based Clustering (HPSO-clustering) for outlier detection. HPSO-clustering uses a partitional approach to generate a hierarchy of clusters. The initial generation consists of the entire swarm. The swarm is then evolved towards a single cluster by merging two clusters of the swarm in each successive generation. For each particle of the swarm, updating of the position is continued until the particle comes to its most suitable position inside the cluster and becomes the true representative of the centroid. The cluster-based approach helps to find outliers based on their distance to the centroids. Less dense data falling at considerable distances from the centroid of the nearest cluster are considered as potential outliers.

Intrusion detection in computer networks is one of the common areas of application of outlier detection. There are some intrusion detection approaches based on ACO, PSO and BCO, and most of them categorize the detected behaviour into normal or abnormal, thus, in a sense, the intrusion detection problem is reduced to a classification or clustering problem[25]. Several examples are given below.

Tsai et al.[26] described an intrusive analysis model based on the design of honeypots and an ant colony; the ACO is applied to trace the trail of attack and analyse the habits and interests of aggressors. Soroush et al.[27] presented one of the pioneering studies, in which ACO is used for intrusion detection, unlike previous approaches in which it was used for intrusion response. Ramos and Abraham were two of the first researchers who attempted to introduce the LF algorithm[28] into the intrusion detection realm[29]. Their model, called ANTIDS, was based on a number of ant-like agents that pick up and drop items with a certain probability to form clusters, distinctively, the cluster with abnormal attributes is typically much smaller in size.

Most of the PSO-based intrusion detection systems (IDS) are hybrid anomaly detection systems, Kolias et al. [25] classified them into: (a) hybrid PSO-Neural Network Systems, (b) hybrid PSO-Support Vector Machine Systems, (c) hybrid PSO-K-means Systems and another category containing intrusion detection systems that employ PSO for the extraction of classification rules.

In addition, Osama et al.[30] proposed an approach using BCO as a search strategy for subset generation, and using Support Vector Machine (SVM) as the classifier based on the selected feature subset, in which the feature selection is one of the prominent factors influencing the quality of intrusion detection systems. In this approach, first, an initial scout bee population is randomly generated, then the fitness of all the bees in the population is measured iteratively. The bees that have the highest fitness are selected as elite bees and the sites visited by them are chosen for neighbourhood search. At the end, SVM is trained based on the best feature subset.

As a whole, SI-based approaches have several advantages over other techniques. More specifically, SI techniques aim to solve complex problems by employing multiple but simple agents without a need for any form of supervision. Every agent collaborates with others toward finding the optimal solution via direct or indirect communication (interactions). These agents are used for discovering classification rules, clusters in anomaly detection[25].

However, to our knowledge, existing SI-based approaches have focused on discovering class level abnormities; and no SI-based approaches for discovering attribute level abnormities in a dataset have been studied. In reality, attribute abnormities happen more frequently, and correcting abnormal attributes rather than eliminating the tuple outliers has advantage of retaining information. Koh et al.[31] gave some statements: class outliers are often the result of one or more attribute outliers, and even when attribute outliers do not affect class memberships, they may still interfere with data analysis; in addition, many real world datasets do not contain class attributes or distinct clusters, and it is meaningful to identify attribute outliers which may be source errors. Thus we will study new approaches based on SI for attribute outlier detection in the following.

4 New ideas on SI approaches for attribute outlier detection

Some studies[9][24] have used attribute correlation analysis to detect attribute outliers in a dataset, but the time complexity of these algorithms is very high. A distribution-based approach[32] eliminates attribute values that do not fit into the distribution models of the dataset, but the accuracy largely depends on the best-fit distribution models used[31].

In this section, we will discuss SI approaches for attribute outlier detection. The main motivation for using SI in the discovery of outliers is that they perform a global search and cope better with attribute interaction, and the time complexity will be reduced.

ID	Name	Gender	City	Province	Country
1	L-199	F	Guangzhou	Guangdong	China
2	L-200	M	Guangzhou	Guangdong	China
3	L-201	M	Nanning	Guangxi	China
4	L-202	F	Guangzhou	***Guandon***	China
5	L-203	M	Nanchang	Jiangxi	China

Table 1: An example of a dataset R.

4.1 Outlier measurement

Rarity is not an indicator of attribute outliers, and observations should be drawn from the correlation behaviour of attributes. Three measures, namely, the O-measure, the P-measure and the Q-measure, are provided in [31]. We use the O-measure in our approach. Related definitions referred to by [31] are described as follows.

Definition 1 (support) Let R be a relation with m attributes $A_1,A_2,\ldots\ldots,A_m$. Let S be a projection on R over attributes $A_u,\ldots, A_v$, i.e. $S = \prod_{Au,\ldots, Av} (R)$. The support of a tuple *s* in S, denoted by sup(*s*), is the count of the tuples in R that have the same values for attributes $A_u,\ldots.A_v$ as item set s.

For example, in Table 1, let $S = \prod_{\text{City, Province}} (R)$, sup(Guangzhou, Guangdong) = 2, sup(Nanning, Guangxi) = 1.

Definition 2 (Neighbourhood) Let tuple *s* = $<a_u,\ldots\ldots, a_v>$. Suppose A_v is the target attribute, the extent of deviation of which we are interested in determining. The neighbourhood of A_v w.r.t *s* is defined as $N(A_v, s) = < a_u,\ldots\ldots, a_{v-1} >$. The support of $N(A_v, s)$ is the count of tuples in R with the same values $a_u ,\ldots\ldots , a_{v-1}$ for $A_u,\ldots.A_{v-1}$.

For example, in Table 1, let $S = \prod_{\text{City, Province}}(R)$, consider tuple *s* = <Guangzhou, Guangdong>, sup(N(Province, *s*)) = 3, sup(N(City, *s*)) = 2.

Definition 3 (O-measure) The O-measure of target attribute A_v w.r.t *s* is defined as

$$\text{O-measure}(A_v, s) = \frac{\sum_{i=u}^{v-1} \text{sup}(N(A_i, s))}{\text{sup}(N(A_v, s))} \qquad (1)$$

For example, let *s* = <Guangzhou, ***Guangdon***, China> be a tuple of $S = \prod_{\text{City, Province, Country}} (R)$. The support of N(Province, *s*) is 3 while sup(N(Country, *s*)) and sup(N(City, *s*)) are both 1. The O-measure of the Province attribute w.r.t. *s* is (1 + 1)/3 = 0.6, i.e. O-measure(Province, *s*) = 0.6.

For comparison, we also compute the O-measure of the Province attribute in tuple *t* = <Guangzhou, Guangdong, China>. We have O-measure(Province, *t*) = (2 + 2)/3 = 1.33.

The lower the O-measure score, the more likely it is that attribute A_v is an attribute outlier in a tuple. In Table 1, O-measure(Province, *s*) is relatively lower than O-measure(Province, *t*), so ***Guangdon*** is more likely an to be attribute outlier.

Consider a relation R with *m* attributes and *n* tuples. In the worst case, scanning all data subspaces (or projected relations) in R requires O($n \times 2^m$) searches, where 2^m is the total number of projected relations on R. Therefore, computing the O-measure scores for each attribute w.r.t every projected relation requires O($2^m \times n \times m$) time complexity[31]. Obviously, the approach of searching all data subspaces of a relation for detecting outliers is highly inefficient. Thus, we make use of a swarm of agents to search heuristically and randomly.

Although different kinds of agents use different search strategies, if they use the same fitness function to evaluate the solutions, they tend to obtain a similar solution set. After stopping searching, the tuples having lower O-measure values (below a threshold) in the solution set include attribute outliers. This does not mean that they are all errors.

4.2 Overview of ACO for detecting attribute outlier

In a relational dataset, each tuple or its projection over some attributes can be represented as a path in a dataset graph (DG). A DG for a dataset with the attribute number m is defined as G = (V, E), where V is a set of all data items, i.e. V = {A = v| A is an attribute name, and v is a value in the domain of A}, and V_i is a subset of V, which only includes data items related to the attribute A_i, $\bigcup_{i=1}^{m} V_i = V$; E is a set of directed edges between two data items in each tuple, i.e. E = {<x, y >| $x \in V_i$, $y \in V_j$, i≠j}.

For example, a dataset is shown in Table 1, and a DG for its projection S=$\prod_{\text{Province, City, Country}}$(R) is given in Figure 1, where

$V = V_1 \cup V_2 \cup V_3$
$= \{v_{11}, v_{12}, v_{13}\} \cup \{v_{21}, v_{22}, v_{23}, v_{24}\} \cup \{v_{31}\}$
= {Guangzhou, Naning, Nanchang} ∪ {Guangdong, Guangxi, Guangdon, Jiangxi} ∪ {China}}

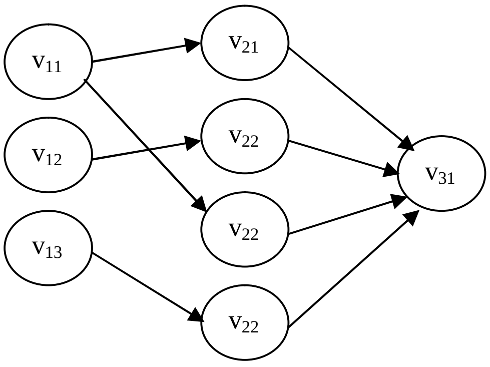

Figure 1: A DG of the dataset projection in Table 1.

For a group of ants, the main steps in detecting attribute outliers by the constructed DG are shown as Algorithm 1.

Some explanations of Algorithm 1 are as follows:

●In step 2), the pheromone of each edge e_i (1≤i≤*Edge_number*) in DG is initialized as Equation (2), where *Edge_number* is the edge number of DG.

$$ {}^{2}{}_{i}(t=0) = \frac{1}{Edge\_number} \tag{2} $$

Algorithm 1: The flow of ACO for detecting attribute outliers

```
1) Set ACO parameters (number of ants, pheromone
   evaporation rate, etc.);
2) Initialize the pheromone trails and create a null outlier
   table S;
3) Assign a group of ants to a start node of DG;
4) Each ant selects a path of DG by the transition rule;
5) Update pheromone trails;
6) Compute the O-measure of attributes from each ant's
   walking path and add or modify outlier information
   into table S;
7) Go to step 4) and repeat until the convergence and
   termination conditions are met.
```

●In step 2), the schema of outlier table S is (T, A, V, M), where T is a tuple ID on an ant traversed path, A is the target attribute name, V is the target attribute value of the tuple, and M is the O-measure of A for the tuple.

●In step 3), each of the start nodes is assigned the same number of ants, and the degree of a start node is zero.

●In step 4), the probability that the adjacent edge e_i is chosen by the ant from node *v* is given by the Equation (3):

$$ P_i(t) = \frac{{}^{2}{}_{i}(t) \cdot \text{¦}_{i}}{\sum_{j} {}^{2}{}_{j} \cdot \text{¦}_{j}, \forall e_j \in I} \tag{3} $$

where

$$ \text{¦}_{i} = \frac{1}{\sup(e_i)} \tag{4} $$

In Equation (3), *I* is the next adjacent edge set of node *v*, ¦_{i} is a heuristic value of edge e_i, v_j and v_k are two vertices of e_i and sup(e_i) is support of (v_h, v_k). From the equation, it can be inferred that the less frequent (v_h, v_k) is, the larger the heuristic value of e_i is.

Pheromones placed on the edges in an ACO system play the role of a distributed long-term memory. This memory is not stored within the individual ants, but is distributed on the edges of the path, which allows an indirect form of communication. This benefits exploitation of prior knowledge. However, it increases the probability of choosing routes belonging to those previously traversed, thus inhibiting the ants from exhibiting a bias toward exploration. In order to enhance the role of exploration, we apply the following transition rule for the ants' selection of the next edge, where *q* is a random number in [0, 1] and ¨ is a parameter in (0, 1).

```
if $q \le \varphi$ then
    select an edge randomly from $I$;
else
    choose $e_i$ with max $P_i$;
end if
```

●In step 5), the pheromones of edges in the newest visited path P are updated by Equation (5), and the pheromones of other unvisited edges are normalized by Equation (6).

$$ {}^{2}{}_{i}(t+1) = (1- {}^{\circ})^{2}{}_{i}(t) + {}^{2}{}_{i}(t)/Q \,, \forall\, e_i \in \mathrm{P} \qquad (5) $$

$$ {}^{2}{}_{i}(t+1) = {}^{2}{}_{i}(t) / \sum_{j} {}^{2}{}_{j} \;, \quad \forall\, e_i \notin \mathrm{P} \qquad (6) $$

In Equation (5), ° in [0,1] is the pheromone evaporation rate, which controls how fast the old visited path evaporates; $Q = \sum \mathrm{Sup}(e_i)$, $e_i \in$ P, such that the smaller Q is, the more the pheromone of the edge in P increases. In Equation (6), $\sum_{j} {}^{2}{}_{j}$ is the sum of the pheromones of all edges.

●In step 6), for the newest selected path P, let the tuple t consist of vertices on P, A is attribute set. Compute the O-measure of each attribute from P, i.e. O-measure(A_i, t), $A_i \in$ A, and obtain the minimum value O-measure(A_k, t) among them; then search the tuple s in table S where N(A_k, s[T]) = N(A_k, t) and s[A] = A_k and s[V]≠t[A_k], and add or modify outlier information into table S by the following rule (call it Rule A).

Rule A:

```
if s exists then
        if O-measure($A_k$, $t$)<$s$[M] then
              $s$[V] = $t$[$A_k$];
              $s$[M] = O-measure($A_k$, $t$);
        end if;
else
    insert ($t$, $A_k$, $t$[$A_k$],O-measure($A_k$, $t$)) into S;
end if.
```

4.3 Overview of PSO for detecting attribute outliers

While ACO solves problems with search spaces that can be represented as a weighted construction graph, PSO solves problems with solutions that can be represented as a set of points in an n-dimensional solution space[1]. We regard each attribute value in a dataset as a point in the 2-dimensional space. The objective is to find the points that can minimize the fitness, i.e. the O-measure. The main steps for detecting attribute outliers in a given dataset by a swarm of particles are shown as Algorithm 2.

Algorithm 2: The flow of PSO for detecting attribute outliers

1) Set PCO parameters (number of particles, positive constants, etc.);
2) Initialize the swarm by assigning each particle to a velocity and a random position;
3) Evaluate the fitness for each particle's position;
4) Update the solution table S and set or reset the best particle;
5) Update the velocities of all particles;
6) Move each particle to its new position;
7) Go to step 3) and repeat until the convergence and termination conditions are met.

Some explanations of Algorithm 2 are:

●In step 2), initialize the velocity of each particle to be zero, i.e. $v_{id} = 0$, $\mathrm{v_{id}}$ represents the velocity of the ith particle in the dth dimension, where i = [1… particle number] and d is 1or 2, where 1 represents the tuple dimension and 2 represents the attribute dimension. At the same time, assign each particle a random position, i.e. p_{id}, which represents the position of the ith particle in the dth dimension, where p_{i1} is [1…tuple number] and p_{i2} is [1…attribute number].

●In step 3), for the ith particle, according to p_{i1} and p_{i2}, we know the particle is at the position of attribute A_k of tuple t, for which the tuple ID is p_{i1} and the attribute ID is p_{i2}, so the O-measure(A, t) can be calculated.

●In step 4), like the solution table used in ACO, the schema of the solution table S is (T, A, V, M), where T is a tuple ID, A is the target attribute name, V is the target attribute value of the tuple and M is the O-measure of A for the tuple. For the ith particle, which is at the position of attribute A_k of tuple t, search the tuple s in table S where N(A_k, s[T]) = N(A_k, t) and s[A] = A_k and s[V]≠t[A_k], add or modify the new information into table S using Rule A presented in section 4.2.

Set or reset the global best particle that has the swarm's best fitness value (i.e. the minimum O-measure value), and let it be the gth particle.

●In step 5), the velocity of each particle is updated by the following equation (7).

$$ v_{id}(t+1) = v_{id}(t) + c_1\, R_1(p_{id}(t) - x_{id}(t)) + c_2\, R_2\, (\mathrm{p_{gd}}(t) - x_{id}(t)), \qquad (7) $$

where

p_{gd} represents the position of the swarm's global best particle in the dth dimension;

$\mathrm{R_1}$ and $\mathrm{R_2}$ are two 2-dimensional vectors with random numbers (each dimension has its own random number) uniformly selected in the range of [0.0, 1.0], which leads to useful randomness for the search strategy;

c_1 and c_2 are positive constant weighting parameters, which generally fall in the range of [0, 4] with $c_1 + c_2 = 4$.

●In step 6), the position of each particle is updated by the following equation shown in (8).

$$ p_{id}(t+1) = |\, p_{id}(t) + v_{id}(t+1)|\ mod\ L_{d,} \qquad (8) $$

where

L_1 is the number of tuples and L_2 is the number of attributes.

4.4 Overview of BCO for detecting attribute outliers

The BCO algorithm has a unique blend of neighbourhood search and random search that makes it appropriate for combinatorial and functional optimizations[33]. For a group of bees, the search area is a 2-dimensional plane, in which each site corresponds to an attribute value of a tuple, denoted as v_{ij}, i.e. the A_j attribute value in tuple t_i. The main steps for detecting attribute outliers in a given dataset by a swarm of bees are shown as Algorithm 3.

Some explanations of Algorithm 3 are:

●In step 1), set parameters, such as the number of scout bees: n, the number of elite bees: e and the number of recruited bees around elite regions: *k*.

Algorithm 3: The flow of BCO for detecting attribute outliers

```
1) Set BCO parameters;
2) Locate each scout bee at a random position;
3) Evaluate the fitness for each scout bee's position;
4) Select elite bees and update the solution table S;
5) Recruit a number of bees around the elite bees for
   neighbourhood search and evaluate their fitness;
6) Select the fittest bee from each elite region;
7) Assign the remaining bees to search randomly and
   evaluate their fitness;
8) Go to step 4) and repeat until the convergence and
   termination conditions are met.
```

●In step 2), *n* bees are arbitrarily located on the search plane, and each site is denoted as L_{ij}, where A_j attribute value in tuple t_i is stored.

●In step 3), for each bee at site L_{ij}, calculate the O-measure of A_j for the tuple t_i, i.e. its fitness.

●In step 4), the elite bees with superior fitness (i.e. the lowest *e* O-measure values) are chosen from *n* bees, and the solution table S is updated using them. Like the solution table used in ACO, the schema of the solution table S is (T, A, V, M), where T is a tuple ID, A is the target attribute name, V is the target attribute value of the tuple, and M is the O-measure of A for the tuple. For the elite bee at L_{ij}, search the tuple *s* in table S where $N(A_j, s[T]) = N(A_j, t_i)$ and $s[A] = A_j$ and $s[V] \neq t_i[A_j]$, add or modify the new information into table S by Rule A presented in section 4.2.

●In step 5), recruit *k* bees around each elite bee for the neighbourhood search and evaluate their fitness. For example, let *k* be 4, the sites for recruited bees around the elite bee at L_{ij} may be $L_{i+1,j+1}$, $L_{i-1,j-1}$, $L_{i+1,j-1}$, $L_{i-1,j+1}$.

●In step 6), select the fittest bee from each elite region, i.e. the best *e* sites.

●In step 7), the remaining *n-e* bees search randomly, and their fitnesses is evaluated.

4.5 The time complexity analysis

Suppose that the number of agents is *k*, the dataset attribute number is *m*, and the tuple number in the dataset is *n*. From the descriptions of the above three algorithms, we can analyse the complexity of the time-consuming process for each algorithm.

Compute the O-measure: $O(nm)$;

Update solution: $O(s)$, where *s* is the changing size of the solution table, and *s* is much less than nm.

If the repeat time is *q*, the time complexity for each algorithm is $O(kqnm)$, which varies linearly with *n* and *m*.

5 Conclusion

This study deals with SI-based approaches in data quality detection. Although SI models have been widely used in many applications, including detection of abnormal behaviour in networks and outliers in databases, limited research has been conducted on attribute outliers. The paper introduces SI models into attribute outlier detection, and presents three approaches based on ACO, PSO and BCO. The algorithms for these approaches are explained in detail. The large searching space is a major bottleneck for detecting attribute outliers. Our SI-based approaches make use of swarms of the fittest agents and random moving agents, and obtain superior solutions by changing the searching paths or the positions of agents heuristically, which improves the time efficiency of outlier detection. Although different search strategies are applied for the three approaches, they use the same fitness function (i.e. the O-measure) to evaluate the solutions. For future study, we intend to perform experiments on large datasets and study a proper set of parameters that can lead to highly accurate results.

Acknowledgement

This work is supported by National Natural Science Foundation of China(U1431227,11403008), Guangdong Scientific and Technological Project (2013B010401017), Natural Science Foundation of Guangdong (S2012010008831).

References

[1] Ahmed, H., Glasgow, J.(2012). Swarm Intelligence: Concepts, Models and Applications, Technical Report 2012-585, School of Computing Queen's University.

[2] Panigrahi, B. K., Shi, Y.H., Lim , M.-H. (2011). Handbook of Swarm Intelligence. Series: Adaptation, Learning, and Optimization, Vol 8, Berlin :Springer-Verlag.

[3] Blum ,C. , Merkle, D. (2008). Swarm Intelligence – Introduction and Applications (Natural Computing serial). , Berlin : Springer.

[4] Yang, X.S., Cui, Z. H.(2014). Bio-inspired computation: success and challenges of IJBIC, International Journal of Bio-inspired Computation, 6(1): 1-6.

[5] Cui, Z. H., Xiao, R.B.(2014). Bio-inspired Computation: Theory and Applications, Journal of Multiple-valued Logic and Soft Computing, 22(3): 217-221.

[6] Yang, X. S., Deb, S., Loomes, M., Karamanoglu, M. (2013). A framework for self-tuning optimization algorithm, *Neural Computing and Applications*, 23(7-8): 2051–2057.

[7] Cui, Z. H., Fan, S. J., Zeng, J. C., Shi, Z.,Z. (2013). Artificial plan optimization algorithm with three-period photosynthesis, *Int. J. Bio-Inspired Computation*, 5(2): 133–139.

[8] Yang, X. S., (2012). Metaheuristics and swarm intelligence in engineering and industry: editorial, Int. J. Bio-Inspired Computation, 4(4): 197–199.

[9] Ciszak, L.(2008) Application of clustering and association methods in data cleaning. Proceedings of the International Multiconference on Computer Science and Information Technology, pp:97-103.

[10] Chiang, F., Miller, R.J.(2008). Discovering data quality rules. Proceedings of the International Conference on Very Large Databases, pp:1166-1177.

[11] Alam, S., Dobbie, G., Riddle, P., Naeem, M.A.(2010). A swarm intelligence based clustering approach for outlier detection, 2010 IEEE Congress on Evolutionary Computation, pp:1-7.

[12] Jindal, R., Sharma, S.D., Manoj Sharma, M.(2013). A New Technique to Increase the Working Performance of the Ant Colony Optimization Algorithm. International Journal of Innovative Technology and Exploring Engineering, 3(2) 128-131.

[13] Beni, G., Wang, J.(1989). Swarm intelligence in cellular robotic systems. Proceedings of NATO Advanced Workshop on Robots and Biological Systems, vol.102, 703-712.

[14] Jiang, K.Q., Song, B.S., Shi, X.L., Song, T.(2012). An Overview of Membrane Computing, Journal of Bioinformatics and Intelligent Control, 1(1): 17-26.

[15] Mohamed Jafar, O.A., Sivakumar, R.(2010). Ant-based Clustering Algorithms: A Brief Survey. International Journal of Computer Theory and Engineering, 2(5) 1793-8201.

[16] Dorigo, M., Caro, G. D. (1999). Ant Algorithms for Discrete Optimization. Artificial Life, 5(3) 137-172.

[17] Dorigo, M., Maniezzo, V. (1996). The ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, 26(1) 1-13.

[18] Dorigo, M., Maniezzo, V., Colorni, A.(1991). Positive feedback as a search strategy, Technical Report 91-016, Dipartimento di Elettronica, Politecnico di Milano, Italy.

[19] Kennedy, J., Eberhart, R. C.(1995). Particle swarm optimization. Proceedings of IEEE International Conference on Neural Networks, 1942–1948.

[20] Aydin, M.E., Wu,J., Zhang,L.(2010) Swarms of metaheuristic agents: A model for collective intelligence, 2010 International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, pp:296-301.

[21] García-Gonzalo, E., Fernández-Martínez, J.L.(2012). A Brief Historical Review of Particle Swarm Optimization (PSO), Journal of Bioinformatics and Intelligent Control, 1(1): 3-16 .

[22] Karaboga, D.(2005). An idea based on honey bee swarm for numerical optimization, Technical Report-TR06,Erciyes University, Engineering Faculty, Computer Engineering Department.

[23] Djenouri, Y., Drias, H., Habbas, Z., Mosteghanemi, H.(2012).Bees swarm optimization for web association rule mining, 2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology,pp:142-146.

[24] Zhu, X., Wu, X. (2004). Class noise vs. attribute noise: A Quantitiative study of their impacts. Artificial Intelligence Review, 22(3) 177-210.

[25] Kolias, C., Kambourakis, G. , Maragoudakis, M.(2011). Swarm intelligence in intrusion detection: A survey. Computers & Security, 30(8) 625-642.

[26] Tsai, C.L., Tseng, C.C., Han, C.C.(2009). Intrusive behavior analysis based on honey pot tracking and ant algorithm analysis. Proceedings of the 43rd Annual 2009 International Carnahan Conference on Security Technology, pp:248-252.

[27] Soroush, E., Saniee A.M., Habibi J.A.(2006). Boosting ant-colony optimization algorithm for computer intrusion detection. Proceedings of The IEEE 20th International Symposium on Frontiers in Networking with Applications.

[28] Lumer, E., Faieta, B.(1994). Diversity and adaptation in populations of clustering ants, Proceedings of Third International Conference on Simulation of Adaptive Behavior: From Animal to Animats 3,pp: 499-508.

[29] Ramos,V., Abraham, A.(2005). ANTIDS: Self organized ant based clustering model for intrusion detection system. Proceedings of The Fourth IEEE International Workshop on Soft Computing as Transdisciplinary Science and Technology(WSTST'05), pp:977-986.

[30] Osama, A., Zulaiha, A.O.(2012). Bees Algorithm for feature selection in network anomaly detection. Journal of Applied Sciences Research, 8(3) 1748-1756.

[31] Koh, J. L. Y. , Lee, M.L., Hsu,W., et al.(2007). Correlation-based detection of attribute outliers. Advances in Databases: Concepts, Systems and Applications(Lecture Notes in Computer Science,vol.4443), 164-175.

[32] Barnett, V., Lewis,T.(1994). Outliers in Statistical Data. New York : John Wiley and Sons.

[33] Pham, D.T., Ghanbarzadeh, A., Koc, E., S. Otri, S., et al. (2006). The bees algorithm—a novel tool for complex optimisation problems. Proceedings of IPROMS 2006 conference, pp:454-461.

Experimental Comparisons of Multi-class Classifiers

Lin Li
Institute of Intelligent Computing and Information Technology, Chengdu Normal University
No.99, East Haike Road Wenjiang District, Chengdu, China
E-mail: lilin200909@gmail.com

Lin Li, Yue Wu and Mao Ye
School. of Computer Science and Engineering, University of Electronic Science and Technology of China
No.2006, Xiyuan Ave, West Hi-Tech Zone, Chengdu, China

Keywords: multi-class classifier, tree classifier, boost classifier, support vector machines, combining boost classifier

Received: June 18, 2014

The multi-class classification algorithms are widely used by many areas such as machine learning and computer vision domains. Nowadays, many literatures described multi-class algorithms, however there are few literature that introduced them with thorough theoretical analysis and experimental comparisons. This paper discusses the principles, important parameters, application domain, runtime performance, accuracy, and etc. of twelve multi-class algorithms: decision tree, random forests, extremely randomized trees, multi-class adaboost classifier, stochastic gradient boosting, linear and nonlinear support vector machines, K nearest neighbors, multi-class logistic classifier, multi-layer perceptron, naive Bayesian classifier and conditional random fields classifier. The experiment tested on five data sets: SPECTF heart data set, Ionosphere radar data set, spam junk mail filter data set, optdigits handwriting data set and scene 15 image classification data set. Our major contribution is that we study the relationships between each classifier and impact of each parameters to classification results. The experiment shows that gradient boosted trees, nonlinear support vector machine, K nearest neighbor reach high performance under the circumstance of binary classification and minor data capacity; Under the condition of high dimension, multi-class and big data, however, gradient boosted trees, linear support vector machine, multi-class logistic classifier get good results. At last, the paper addresses the development and future of multi-class classifier algorithms.

Povzetek: V prispevku je podan pregled klasifikatorjev z več razredi.

1 Introduction

Multi-classification problem [1] is that of supposing a set of training data $(x_1,c_1),...,(x_n,c_n)$, where $x \in R^p$ are finite set of input features, $c_i \in \{1,2,...,K\}$ are class numbers of output variables. The purpose of multi-classification task is to find a classifying rules based on the training sample, then given a new features, outputting a classifying category. Today multi-classifier algorithms are applied to a variety of application areas such as: radar signal classification, character recognition, remote sensing, medical diagnostics, expert systems, voice recognition domains and etc.

Multiple classifiers has a long history. Selfridge et al. [2] first propose a multi-classification system based on ‘winners get all’ solution which chooses the optimal solution as a multi-classifier output. Kanal and Minsky [3, 4] play an important role in multi-classifiers development. They claim that any classifying algorithm does not solve all problems. We need to design specific classifying algorithm for different problems. Multilayer perceptron [5] is an artificial neural network model that can resolve this kind of nonlinear data. Decision tree is an ancient non-parametric classification algorithm that classifies the samples according to the classifying rules. Leo Breiman[6] proposes random forest as a good solution to the scalability issues of single decision tree. Adboost algorithm is proposed by Yoav Freund and Robert Schapire[34] is a meta-classification algorithm which can be combined with other classification algorithms to enhance its performance. Multi-class logistic classifier proposed by Jerome Friedman et al. [7] is another important improvement of enhancing the basic boosting algorithm. K nearest neighbor [5] classifies samples based on the adjacent spatial relationships of features. In 1980s with the rise of data fusion and learning model in statistics and management science, Bayesian expert [8-10] system is widely used. Since the 1990s, Vapnik proposes support vector machines, transforming the feature from low dimensional space into high dimensional space, which is a better ways to classify the features. Nonlinear support vector machine gets a great success, however it is not ideal in some cases i.e. the original features are already high dimensional space, so people propose a linear support vector machine[11] for these cases. Because of the complexity of the data, a single classifier is often difficult to obtain good performance for specific applications, it is a growing tendency to improve classifying performance by a combination of classification methods [12].

How to solve the multi-classification problems is challenging. There are two ways to deal with it [13]. Nilsson et al. [14] first use combination of binary classifiers to solve the multi-classification problems. The other way is that directly extends binary classifier into multiple classifier.

The main contributions of this paper are as following:

1. We compare twelve most commonly used algorithms for multi-classification in several aspects such as principle, important parameters, running time performance, and etc.

2. This paper considers the impact of differences of type and size in data sets. We chose binary classification and multi-classification, a small amount of data and a large size set as the evaluation data sets.

3. This paper gives in-depth analysis of multi-category classification for each class.

4. In this paper, we investigate the relationship between the single classifier and a combination of single classifier.

This paper discusses the 12 multiple classifiers: decision trees, random forests, extremely randomized trees, multi-class adaboost, stochastic gradient boosting, support vector machines (including linear and nonlinear), K nearest neighbors, multi-class logistic classifier, multilayer perceptron, naive Bayesian classifier and conditional random fields classifier.

The paper is organized as follows: section 1 is an introduction of studying content. In section 2 we discuss the related works. Overview of algorithms are discussed in section 3. Section 4 presents an experimental setup and parameters settings. In Section 5, we explain experimental results. Section 6 concludes the paper.

2 Related works

There are few comprehensive comparisons of multi-classification algorithms. King et al. [15] is the most comprehensive and earliest study of multi-classification algorithms including CART, the traditional algorithm C4.5, Naïve Bayes, K nearest neighbor, neural networks, and etc. However after that a few emerging algorithms such as support vector machines, random forests have been widely used. Further, data sets they used are too small while comparing to current big data. Then again their evaluation criteria is simple. Bauer et al. [16] thoroughly compare voting classification algorithms including bagging, boosting and its improved versions, but it is only comparing these two types of voting algorithm. LeCun et al. [17] use accuracy, rejecting rates and running time as the evaluation criteria. They compare algorithms: K-nearest neighbor, linear classifier, the main ingredient and polynomial classifiers on handwritten recognition data set. But only one data set used, it is not sufficient to evaluate different application scenarios. Ding et al. [18] use neural networks and support vector machines for protein test data set. They do detailed comparison of the accuracy of different parameters, but this comparison is relatively simple and data set is small. Tao et al. [19] study the decision trees, support vector machines, K nearest neighbor classification of gene sequences of organization application. However, this comparison only discusses single dataset. Foody et al [20] study multi-class image classification by support vector machines. But they only compare support vector machines, even linear support vector machine is not involved. Chih-Wei Hsu et al. [21] also study multi-class support vector machines for a more in-depth theoretical analysis and comparison, but is limited to multi-class support vector machine. Caruana et al. [22] study supervised learning algorithm (support vector machines, neural networks, decision trees, and etc.) in 9 different criteria such as ROC area, F evaluation and etc., but the literature is only discussed two classification data sets. Krusienski et al. [23] compare the Pearson correlation method, Fisher linear discriminant, stepwise linear discriminant, linear support vector machines, and Gaussian kernel support vector machines on P300 Speller data set. However, the data set is relatively simple, and small size of data is often difficult to compare running performance of support vector machine between linear and non-linear scenarios.

Table 1 lists the current situation of classifier comparisons.

Reference	Comparison of algorithms	Data sets	Evaluation criteria	Research domains
King et al.[15]	Symbolic learning(CART, C4.5, New ID, AC^2, Cal5, CN2) , statistic learning(Bayesian network, K-nearest, kernel density, linear discrimination, quadratic discrimination , logistic regression) , neural network	Satellite image, hand written digits, vehicle, segment, Credit risk, Belgian data, Shuttle control data, Diabetes data, Heart disease and head injury, German credit data	Accuracy	General purpose
Bauer et al[16	Bagging, boosting and its variants	Segment, DNA, chess, waveform, sat-image, mushroom, nursery, letter, shuttle	Average error rate, variance, bias	General purpose
Ding et al.[18	Support decision vector, neural network	Protein test dataset	Accuracy, Q-percentage, Accuracy	Bioinformatics
Tao et al.[19]	Support decision vector, Bayesian network, K-nearest, decision tree	ALL, GCM, SRBCT, MLL-leukemia, lymphoma, NCI60, HBC	Accuracy	Bioinformatics
Foody et al.[20]	Support decision vector, decision tree, discriminating analysis, neural network	Airborne sensor data	Accuracy,	Remote imaging
Chih-Wei Hsu et al[21]	Support decision vector	Iris, wine, glass, vehicle, segment, DNA, sat-image, letter, shuttle	Accuracy, running-time	General purpose

Caruana et al.[22]	Support decision vector, neural network, decision tree, memory based learning, bagged tree, boosted tree, boosted stumps	RMS, MXE, CAL, ADULT.	Accuracy, square error, inter-class cross entropy, ROC regions, F evaluation, recall and precision, average precision and recall, lift, probability calibration	General purpose
Krusienski et al.[23]	Pearson related methods, Fisher linear discrimination, stepwise linear discrimination, linear support decision vector, Gaussian kernel support decision vector	P300 Speller	ROC curve	Medicine domain
Our method	Decision tree, random forests, extremely randomized trees, multi-class adaboost classifier, multi-class logistic classifier, stochastic gradient boosting, multilayer perceptron, K nearest neighbors, naive Bayesian classifier and support vector machines(including linear and nonlinear)	SPECTF, Ionosphere, spam, optdigits and Scene 15	Overall accuracy, average precision and recall, average Jaccard, inter-class F and Jaccard evaluation, running performance.	General purpose

Table 1: Comparisons of multi-class classification algorithms.

Table 1 shows that the majority of current surveys focus on multi-class classifiers in a particular field, such as medicine, biology, remote sensing images, and etc. The data sets and methods used for evaluation is relatively simple. King et al's evaluation [15] is more comprehensive, however the comprising algorithms are classical. After that new algorithms emerge. Our comparing algorithms are the newest and most representative of the current tendency in a variety of application domains.

3 Overview of Algorithms

3.1 Brief introduction of algorithms

In order to better understand various classifiers to compare, we briefly introduce multiple classifier algorithms.

3.1.1 Decision tree

In machine learning, decision trees [24] is a predictive model. A decision tree is a flowchart-like structure in which internal node represents test on an attribute, each branch represents outcome of test and each leaf node represents class label, and then decision is taken after computing all attributes. A path from root to leaf represents classification rules. Decision tree is actually an adaptive basis function model [25], and can be expressed as follows

$$f(x)=E[y\,|\,x]=\sum_{m=1}^{M} w_m I(x\in R_m)=\sum_{m=1}^{M} w_m \text{¤}(x;v_m) \tag{1}$$

Where R_m is the $m'th$ region, w_m is the mean response in this region, and v_m encodes the choice of variable to split on, and the threshold value, on the path from the root to the $m'th$ leaf.

Classification tree is an ancient method, it has various variants, typically such as[26] ID3 (Iterative Dichotomiser 3) proposed by Ross Quinlan, is a greedy approach that in each iteration choose the best attribute value to split the data, but this method has the problem of local optimum. C4.5 also proposed by Ross Quin lan, is an improved ID3, can be used for classification. CART(classification and regression Trees)[27] proposed by Breiman has same process with C4.5 algorithm, except that the C4.5 uses an information entropy rather than Geni coefficient used by CART.

3.1.2 Random forests

Random forests are proposed by Leo Breiman and Adele Cutler[28]. It is an ensemble learning method for classification (and regression) that is built by constructing a multitude of decision trees at training time and outputting the class by voting of individual trees.

Random forests are a method of building a forest of uncorrelated trees using a CART like procedure, combined with randomized node optimization and bagging [29].

Random forests have the advantages of computing efficiency, improving the prediction accuracy without significantly increase of computational cost. Random forest can be well predicted up to thousands of explanatory variables [30], known as one of the best current algorithms [31].

3.1.3 Extremely randomized trees

Extremely randomized trees have been introduced by Pierre Geurts, Damien Ernst and Louis Wehenkel[32]. The algorithm of growing extremely randomized trees is similar to random forest, but there are two differences:

1. Extremely randomized trees don't apply the bagging procedure to construct a set of the training samples for each tree. The same input training set is used to train all trees.

2. Extremely randomized trees pick a node split very extremely (both a variable index and variable splitting value are chosen randomly), whereas random forests find the best split (optimal one by variable index and variable splitting value) among random subset of variables.

3.1.4 Multi-class adaboost classifier

Boosting has been a very successful technique for solving the two-class classification problem [33]. In going from two class to multi-class classification, most boosting algorithms have been restricted to reducing the multi-class classification problem to multiple two-class problems [34].

3.1.5 Stochastic gradient boosting

Gradient boosting proposed by Friedman [35] is a method to improve basic boosting algorithm. The traditional boosting method is adjusted weights to correct classification samples and error samples based on gradient descend at each iteration. The major difference of gradient boosting from the traditional boosting method is that purpose at each iteration is not to reduce the losses, but in order to eliminate the loss. The new model at each iteration is based on the residuals of former process. Inspired by Breiman[6]'s randomized bagging idea, Friedman introduces stochastic gradient boosting by randomized down-sampling to train basic classifier.

3.1.6 Support vector machines

Support vector machines [36, 37] is the method that mapped feature vector into a higher dimensional vector space, where a maximum margin hyper-plane is established in this space. So we choose the hyper-plane so that the distance from it to the nearest data point on each side is maximized. The greater the distance between the nearest data of different classes is, the smaller the total error classification is. The multi-class support vector machines [21] can be defined as follows

Supposing l groups sample: $(x_1,c_1),\dots,(x_l,c_l)$, where $x_i \in R^n, i=1,\dots,l$, $c_i \in \{1,\dots,k\}$ is the type of x_i . The $i-th$ support vector machine solve this problem.

$$\min_{w^i,b^i,\P^i} \quad \frac{1}{2}(w^i)^T w^i + C\sum_{j=1}^{t} \P_j^i (w^i)^T \tag{2}$$

$$(w^i)^T \text{¤}(x_j) + b^i \ge 1 - \P_j^i, if \;\; c_j = i \tag{3}$$

$$\P_j^i \ge 0, \quad j = 1,\dots,l \tag{4}$$

Where training sample x_i is mapping into high dimensional space by function ¤ and regularization parameters C . Minimizing $(1/2)(w^i)^T w^i$ means that we have to minimize $2/\|w^i\|$, the margins between two group data. The penalty $C\sum_{j=1}^{t}\P_j^i(w^i)^T$ is to reduce the number of training error. The core concept of support vector machine is to seek a balance between the regularization term $(1/2)(w^i)^T w^i$ and the training errors.

We get k decision functions after solving formula (4) .

$$\begin{gathered} (w^1)^T \text{¤}(x) + b^1 \\ \vdots \\ (w^k)^T \text{¤}(x) + b^k \end{gathered} \tag{5}$$

After that, we have the largest value of decision functions as the predictive class of x .

3.1.7 Linear support vector machines

SVM uses a nonlinear mapping that converts low-dimensional feature space into a high-dimensional space to get better discriminative. However, in other applications, the input feature itself is a high-dimensional space, if more is mapped to more high latitudes, it may not be able to get better performance. Their own space can be directly used as identification. The linear support vector machine SVM [38] is suitable for this scenario. For multi-classification, Crammer et al. [39] propose this method to solve the problem. We define the original question as follow

Supposing the training data set $T = \{(x_1,c_1),(x_2,c_2),\dots,(x_N,c_N)\}$, where, $x_i \in \text{¡} \subseteq R^n$ is feature vector, $y_i \in Y = \{1,2,\dots,k\}$ is the type of instance, $i = 1,2,\dots,l$. The multi-class problem can be formulated as the following primal problem.

$$\begin{aligned} &\min_{\{\mathrm{w}_m\},\{\P_i\}} \frac{1}{2}\sum\nolimits_m \|w_m\|^2 + C\sum\nolimits_i \P_i \\ &s.t. \quad w_{y_i}^T x_i \ge e_i^m - \P_i \quad \forall_m, i, \end{aligned} \tag{6}$$

where, $C > 0$ is the regularization parameter, w_m is the weight vector associated with class m , $e_i^m = 1 - \pm_{c_i,m}$, and $\pm_{c_i,m} = 1$ if $c_i = m$, $\pm_{c_i,m} = 0$ if $c_i \ne m$. Note that in (7), the constant $m = c_i$ corresponds to the non-negativity constraint: $\P_i \ge 0$. The decision function is

$$\arg\max_m w_m^T x \tag{7}$$

3.1.8 K nearest neighbors

K nearest neighbours algorithm (or k-NN for short) is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space.

In k-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbours, with the object being assigned to the class most common among its k nearest neighbours (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbour.

3.1.9 Multi-class logistic classifier (maximum entropy classifier)

In some cases, multi-class logistic regression well fits features. It has the formula [25]

$$p(y = c \mid x, W) = \frac{\exp(w_c^T x)}{\sum_{c'=1}^{C} \exp(w_{c'}^T x)} \tag{8}$$

Where, $p(y = c \mid x, W)$ is the predictive probability. y is the class type of totally C . w_c is the weight of class c, and approximated by maximum posterior probability. With this, the log-likelihood has the form

$$\begin{aligned} l(W) &= \log \prod_{i=1}^{N} \prod_{c=1}^{C} \text{«}_{ic}^{y_{ic}} = \sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic} \log \text{«}_{ic} \\ &= \sum_{i=1}^{N} \left[\left(\sum_{c=1}^{C} y_{ic} w_c^T x_i \right) - \log \left(\sum_{c'=1}^{C} \exp(w_{c'}^T x_i) \right) \right] \end{aligned} \tag{9}$$

Where, $\text{«}_{ic} = p(y_i = c \mid x_i, W)$. This model can be optimized by L-BFGS[41].

3.1.10 Multilayer perceptron

A multilayer perceptron (MLP) is a feed forward artificial neural network model that maps sets of input data onto a set of appropriate outputs. A MLP consists of multiple layers of nodes in a directed graph, with each layer fully connected to the next one. Except for the input nodes, each node is a neuron (or processing element) with a nonlinear activation function.

MLP[57] has evolved over the years as a very powerful technique for solving a wide variety of problems. Much progress has been made in improving performance and in understanding how these neural networks operate. However, the need for additional improvements in training these networks still exists since the training process is very chaotic in nature.

3.1.11 Naive Bayesian classifier

A naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions. A more descriptive term for the underlying probability model would be “independent feature model”

Assuming sample X , belongs to type C_i . The class-conditional density is

$$P(C_i \mid X) = \frac{f(X \mid C_i)P(C_i)}{p(X)} = \frac{f(X \mid C_i)P(C_i)}{\sum_{j=1}^{n} f(X \mid C_j)P(C_j)} \tag{10}$$

Where, $f(.|C_j)$ is the maximum likelihood. Input features is x , c is class type.

A simpler alternative to generative and discriminative learning is to dispense with probabilities altogether, and to learn a function, called a discriminant function, that directly maps inputs to outputs. The decision function of naive Bayesian classifier is

$$c = \arg\max_{c_k} P(\mathrm{C} = c_k) \prod_{j=1}^{n} P(\mathrm{Y}_i = \mathrm{y}_{\mathrm{j}} \mid \mathrm{C} = c_k) \tag{11}$$

3.1.12 Conditional random fields classifier

CRFs (Conditional random fields) are a class of statistical modelling method often applied in pattern recognition and machine learning, where they are used for structured prediction. Whereas an ordinary classifier predicts a label for a single sample without regard to "neighboring" samples, a CRF can take context into account; e.g., the linear chain CRF popular in natural language processing predicts sequences of labels for sequences of input samples.

We define a CRF on observations X and random variables Y as follows:

Let $G = (V , E)$ be a graph such that $Y = (\mathrm{Y}_{\mathrm{v}})_{v \in V}$, so that Y is indexed by the vertices of G . Then $(\mathrm{X,Y})$ is a conditional random field when the random variables Yu, conditioned on X, obey the Markov property with respect to the graph:

$$p(\mathrm{Y}_{\mathrm{v}} \mid \mathrm{X}, \mathrm{Y}_{\mathrm{w}}, \mathrm{w} \neq \mathrm{v}) = \mathrm{p}(\mathrm{Y}_{\mathrm{v}} \mid \mathrm{X}, \mathrm{Y}_{\mathrm{w}}, \mathrm{w} - \mathrm{v}) \tag{12}$$

Where $w \square v$ means that w and v are neighbors in G. What this means is that a CRF is an undirected graphical model whose nodes can be divided into exactly two disjoint sets X and Y , the observed and output variables, respectively; the conditional distribution $p(\mathrm{y} \mid \mathrm{x})$ is then modeled. For classification problem, we compute the maximum conditional probabilistic distribution.

3.2 Comparison of algorithms

With reference to [45] for the multi-classification, we made a comparison of 12 algorithms as shown in table 2.

We analyze and summarize in Table 2 as follow:

1. Algorithms type

Aside from Naive Bayes, others are discriminant model. Bayesian algorithm by learning the joint distribution $P(C,X)$, then obtains the conditional probability $P(X \mid C)$. The classifying prediction is achieved by maximizing likelihood approximation. However the discriminant method directly makes prediction by the discriminant function or conditional probability.

2. Algorithms trait

Decision trees, random forest, extreme random tree, multi-class adaboost upgrade and stochastic gradient boosting all belong to model with adaptive basis functions that can be grouped into common additive model[25], as shown in Equation 13.

$$f(x) = \sim + f_1(x_1) + \ldots + f_D(x_D) \tag{13}$$

Where f_i is the sub-model obtained through training sample. $f(x)$ is the superposition of sub-models. Decision tree is the basic sub-model for tree-like algorithms. Upon whether the use of all samples for training, these kinds of algorithms can be divided into random forests and extremely random tree. Random forests is to build sub-model through random bagging sampling. However, extremely random tree obtains sub-model using all training samples, but randomly selecting the splitting threshold. Multi-class adaboost and stochastic gradient boosting is a linear combination of sub-models (weak classifiers). The difference lies in their learning algorithms.

Multilayer Perceptron, linear and non-linear support vector machines can be classified as kernel methods [46]. The unified formula has the form

$$f(x) = w \cdot \Phi(x) + b \tag{14}$$

where w is real weight , $b \in R$ is bias. $\Phi(x)$ function is the type of the classifiers, for MPL $\Phi(.) = (\Phi_1(\cdot), \Phi_1(\cdot), \ldots, \Phi_N(\cdot))$, the $i-th$ hidden is defined as $\Phi_i(x) = h(v_i x + d_i)$. h is the mapping function , generally a hyperbolic function or shape function B is chosen by MPL. For linear support vector machine $\Phi(x)$ is linear function, rather than polynomial function, Gaussian kernel function, and etc. for non-linear support vector machine.

K nearest neighbor model is constructed according to the division of distance relationship of the training feature space, being classified by a majority voting.

Multi-class logistic regression (maximum entropy) is the probabilistic choice model with constraints that uncertain contents are treated with equal probability of using entropy maximization to represent.

Naive Bayes classifier is based on the conditional independence assumption of training samples, learning parameters with the joint probability distribution though Bayes' theorem, then classifying a given sample, by maximizing the posterior probability to get corresponding class.

3. Learning policy, loss and algorithms

Decision trees, random forests, extremely randomized trees belong to the maximum likelihood approximation of learning strategies, with the loss of log-likelihood function.

Algorithms	Algorithms type	Algorithms characteristic	Learning policy	Loss of learning	Learning algorithms	Classification strategy
Decision tree	Discriminant	Classification tree	Regularized maximum likelihood estimation	Logarithmic likelihood loss	Feature selection, generation, prune	IF-THEN policy based on tree spitting
Random tree	Discriminant	Classification tree(based on bagging)	Regularized maximum likelihood estimation	Logarithmic likelihood loss	Building multi-decision tree based on bagging of subsampling	Sub-tree voting
Extremely random tree	Discriminant	Classification tree	Regularized maximum likelihood estimation	Logarithmic likelihood loss	Building multi-decision tree	Sub-tree voting
Multi-class adaboost	Discriminant	Linear combination of weak classifier(based on decision tree)	Addition minimization loss	Exponent loss	Forward additive algorithm	Linear combination of weighted maximum weak classifiers
Stochastic gradient boosting	Discriminant	Linear combination of weak classifier(based on decision tree)	Addition minimization loss	Exponent loss	Stochastic gradient descent algorithm	Linear combination of weighted maximum weak classifiers
Non-linear Support vector machine (based on libsvm)	Discriminant	Super-plane separation, kernel trick	Minimizing the loss of regular hinge, soft margin maximization	Hinge loss	Sequential minimal optimization algorithm (SMO)	Maximum class of test samples
Linear SVM (based on liblinear)	Discriminant	Super-plane separation	Minimizing the loss of regular hinge, soft margin maximization	Hinge loss	Sequential dual method	Maximum weighted test sample
K-nearest	Discriminant	Distance of feature space				Multiple voting, empirical loss minimization
Multi-logistic (Maximum entropy)	Discriminant	Conditional probabilistic distribution, Log-linear model	Regularized maximum likelihood estimation	Logistic loss	L-BFGS	Maximum likelihood estimation
Multilayer perceptron	Discriminant	Super-plane separation	Minimization of error separation distance point to the hyper-plane	Error separation distance point to the hyper-plane	Random gradient decrease	Maximum weighted test sample
Naive Bayesian classifier	Generative	Joint distribution of feature and class, conditional independent assumption	Maximum likelihood estimation, Maximum posterior probability	Logarithmic likelihood loss	Probabilistic computation	Maximum posterior probability
Conditional Random Fields	Discriminant	Conditional probabilistic distribution under observing sequence, Log-linear model	Maximum likelihood estimation, Regularized maximum likelihood estimation	Logarithmic likelihood loss	Random gradient decrease, quasi-newton methods	Maximum likelihood estimation

Table 2: Summary of twelve multi-class methods.

The decision tree’s optimal strategy is to learn some features through a recursive process and split the tree in accordance with the feature of the training samples. In order to have a better generalization ability, the decision tree model has to be pruning for removal of over-fitting.

Random forests is based on random sampling based on the approach (bagging) to form more stars forest trees. Extremely randomized trees is randomly selecting the splitting value to build decision trees forests.

Multilayer Perceptron, linear and non-linear support vector machines are to separate hyper-plane. The difference is that multilayer perceptron is to minimize the error hyper-plane, however linear and non-linear support vector machine is a minimal loss of hinge page. Perceptron learning algorithm is stochastic gradient descent, linear support vector machine is a sequential dual method, and non-linear support vector machine is a sequential minimal optimization method.

K-nearest neighbor is based on distance of feature space.

Multi-class logistic classifier (maximum entropy) learning strategies can be seen as either maximum likelihood estimation, or a logical loss minimization. Loss of function is a linear logarithmic function. The model learning is the maximum likelihood estimation or regularized maximum likelihood estimation under certain training conditions. Optimization algorithm is L-BGFS.

Naive Bayesian probability is a typical generative model. The maximum likelihood estimation and Bayesian posterior probabilities is calculated based on the joint probability and conditional independence assumption.

4. Classification strategy

Decision tree uses the IF-THEN rules for classification decisions based on the value of the model learned. Random forests and extremely random tree is based on the voting of every single decision tree classification, then taking the highest vote as the final results.

Multilayer Perceptron, multi-class logistic regression (maximum entropy), linear and non-linear support vector machine have the same form of classification decisions

$$class\ \ of\quad x \equiv \arg\max_{i=1,\ldots,k}((w^i)^T ¤(x) + b^i). \tag{15}$$

The difference lies in the choice of $¤(x)$. Multilayer perceptron machine chooses B-shaped function, hyperbolic tangent function, and etc.; Log-linear functions is chosen for multi-class logistic regression; linear support vector machine choses a linear function; nonlinear support vector machine's choice is non-linear kernel function.

K nearest neighbor is a majority voting that output classification is determined by choosing K nearest voting in light of test sample' s distance to the learned model.

Naive Bayesian decision strategy is the rule of maximizing the posterior probability.

4 Experimental Setup

In order to evaluate the performance of various types of classifiers, we implemented our comparisons based on Darwin [47], Opencv[56], Libsvm [48] and liblinear [11] in C++. This paper compared 12 kinds of algorithms.

4.1 Performance comparisons

Confusion matrix (Confusion Matrices) [49, 50] is a common performance evaluation method in pattern recognition classification, which characterizes the relationship between the type of real classes and the recognition classes. For multi-classification problem (For simplifying the representation, we take three categories as example) is illustrated in Table 2.

Prediction class True class	A	B	C
A	AA	AB	AC
B	BA	BB	BC
C	CA	CB	CC

Table 3: Statistics of confusion matrices for samples classification.

Where A, B and C are three classes, AA, BB and CC represent the correct prediction number of samples, the remaining number of samples is representative of the error prediction. AA represents the number of samples correctly identified as samples A. AB is predictive number that original Sample A which is incorrectly predicted as Sample B. The remaining items have the same meaning.

Total accuracy rate can be calculated based on confusion matrix as follows.

$$TA = (AA + BB + CC) / (AA + AB + ... + CC) \tag{16}$$

Where $AA + AB + ... + CC$ is the total number of sample. TA is the total accuracy.

Precision and recall are quantitative evaluation method. They are not only used to evaluate the accuracy of each class, but also the important standard to measure the performance of the classification system.

Precision is the fraction of retrieved instances that are relevant. Precision reflects the classification accuracy. In practical applications, the average precision are often used to evaluate multi-classification (taking categories as example), which is calculated as follows

$$avgprecision = (AA / (AA + BA + CA) + BB / (BA + BB + BC) + CC / (AC + BC + CC)) \tag{17}$$

Recall is the fraction of relevant instances that are retrieved. Recall reflects the classification comprehensiveness. In practical applications, the average recall are often used to evaluate multi-classification (taking categories as example), which is calculated as follows

$$avgrecall = (AA / (AA + AB + AC) + BB / (BA + BB + BC) + CC / ((CA + CB + CC)) \tag{18}$$

F_1 Measure is an integrated measurement method of the recall and precision. Higher values reflect the recall and precision better integrated. F_1 is defined as

$$F_1 = 2 * \frac{avgprecision * avgrecall}{avgprecision + avgrecall} \tag{19}$$

Jaccard Coefficient [51] is used for comparing the similarity and diversity of sample sets. Taking class A as example, the formula is

$$JC_A = \frac{AA}{AA + AB + AC + BA + CA} \tag{20}$$

Average Jaccard coefficient reflects the average of the various categories Jaccard coefficient, which is calculated as

$$avgJC_A = (\frac{AA}{AA+AB+AC+BA+CA} + \frac{BB}{AB+BB+CB+BA+BC} + \frac{CC}{CA+CB+CC+AC+BC})/3 \tag{21}$$

Jaccard coefficient predicts more accurately reflects higher. Jaccard coefficient can evaluate multi-class classification in each class.

4.2 Data sets and data transformation

We evaluate the performance of these algorithms on five data sets that consists of three binary classification data sets and two multi-classification data sets.

4.2.1 SPECTE Heart data set

SPECTF[52] means single-photon emission computed tomography cardiac data sets. SPECTF is a new data set[53]. In cleaning process, the records with missing information, incomplete picture records are filtered, since the original image scale is not uniform, so the original image is transformed into gray image (0 to 255).

After cleaning, the data set contains 44 features that record 22 region of interest for the cardiac systolic and expanded state. The data type is an integer type (rang from 0 to 100). The data set has total 267 instances of which containing 80 training samples and 187 test samples. Each instance has two states: normal and abnormal. In experiment, we converted class label into class 0 (normal) and class 1 (abnormal). Histogram distribution of training samples showed normal shape is shown in Figure 1 (a).

4.2.2 Ionosphere data set

Ionosphere data set[52] includes is a set of radar data sets created by a military system acquisition in Labrador NATO airbase, Goose Bay, Canada.

The data set has total of 34 feature (integer, decimal type) that record 17 sub-pulses labels and values. The data set consists of 351 instances of which 100 instances are training samples, 251 instances are test samples. Each instance has two states: good and bad. In the experiment we converted class label into class 0 that represents the existence of the facts) and class 1 the non-existent facts. Feature type of the dataset is real type, range from -1 to

a. Histogram of the distribution of the training samples is shown in Figure 1 (b).

4.2.3 Spam data set

Spam[52] is a spam filtering data set that has total of 57 data features. 48 consecutive real number is used to represent the percentage of messages of 48 words (i.e. make, address, all, conference) and 6 word occurrences (!, (, [,, $, #) in email.

The data set has a total of 4601 instances, where the training instances are 3065, testing instances are 1,536. Each instance has two states: 0 for not spam and 1 for junk mail. The data set is real type feature information, the range is 0 to 9090. The training sample distribution histogram is scattered, as shown in Figure 1 (c).

4.2.4 Optidigits data set

Optdigits[52] is the information collection extracted from 43 individuals (30 of them for training, 13 as a testing) by the handwritten Optical Character Recognition bit image Standardization American National Standards Institute of Technology (NIST).

The data set has a total of 5620 instances, where the training instances are 3823, test instances are 1797. The instances have ten classes, from 0 to 9 digits. Feature data type is a positive integer, ranging from 0 to 16. Histogram distribution of training samples is shown in Figure 1 (d).

4.2.5 Scene 15 data set

Scene 15[54] data set is a post-processing data set. The original data set has total of 15 classes, 4485 images. We randomly selected sample of 2250 as training samples, 2235 samples for testing.

We used the method of PHOW (Pyramid Histogram Of visual Word) [54] for feature extraction. 12000 descriptors were obtained for each image. In order to obtain a better classification within the features, we used method proposed by vedaldi[55] for features kernel transformation, finally we got 36,000 dimensional feature descriptors to characterize a single image. In the experiment we converted 15 classes into class 0 to 14, representing the bedroom, CALsuburb, industrial, kitchen, livingroom, MITcoast, MITforest , MIThighway, MITinsidecity, MITmountain, MITopencountry, MITstreet, MITtallbuilding, PARoffice and store respectively. Feature data type is a positive real number, the range is -0.3485 to 0.4851, further feature data is relatively concentrated, similar to normal training. Histogram distribution of training samples is shown in Figure 1 (e).

This data set has large dimensions of features and a large size of data. Selection of this data set is mainly to test algorithms adaptability for real application scenarios.

In Figure 1, we demonstrated our diversity of data sets. SPECIF, ionoispher and spam are small scale data sets. They are for binary classification. Optigits and Scene 15 are large scale data sets. They are for multiclass classification. At the same time, the range of our feature vector value are large.

The range of Figure 1 feature values is also huge. The range of SPECIF is from 0 to 100, ionoispher from -1 to 1, and spam from 0 to 1000, Optigits from 0 to 20 and Scene 15 from -0.5 to 0.5. The diversity for our comparing experiments are necessary. This shows that our experiments are rich and valuable.

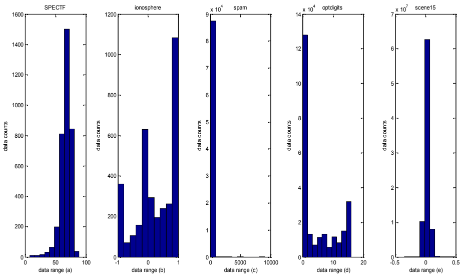

Figure 1: The histogram characteristics of training data sets.

4.2.6 Dataset selection

We select the data set is based on the following considerations.

1. Both binary classification and multi-classification data sets were taken into account for the purpose that some algorithm are superior to binary classification, while others are more suitable for multi-classification problems.

2. For better evaluation of the adaptability of the algorithms, we chosen both low dimensional data set with small size and high dimensional data set with large size.

3. We also paid attention to the data type differences, both pure integer and double data type were included. There were also real data type features, which had either wholly positive or negative features.

4. It can be seen in Figure 1 that the histogram of the distribution of the data we have chosen the difference is quite different. Some are concentrated, just like normal distribution, while others are relatively sparse.

4.3 Parameter setting and selection

4.3.1 Decision tree

1. The maximum depth of tree: The default value was set to 1. With the value increase, classification accuracy and running time will increase. We set maximum depth to 2, 4, 6, 8, 10, 12, 14, 16, 18 and 20 respectively. We found the highest accuracy rate when the maximum depth is 10.

2. Splitting Criteria Consideration: We tested the wrong classification rule, entropy rule, and Geni coefficient rule respectively. We found that Geni coefficient got the best of all, and mistake classification rule got the worst of all.

3. The first split minimum number of samples were tested with value of 0, 2, 4, 6, 8 and 10 respectively. We found the minimum number of samples was increased, but performance degraded.

4. The maximum number of features of the training sample, the default was set to 1000.

4.3.2 Random forests

1. A maximum depth of random forests, as the same of decision tree, the optimal value was set to 10.
2. The maximum number of training samples feature, the default was set to 1000, the value should change with the number of sample features change, as long as the sample itself to adapt to the largest number of features.
3. The number of decision trees: we tested the value of 20, 40, 60, 80, 100, 120, 140, 160, 180 and 200 respectively. We found that the number increases, the execution time also increases, and after over the value of 100, the improvement of the accuracy rate is not obvious. So we set it to 100.
4. The accuracy of the random forest was used to control the iteration. We tested the value of 0.0001, 0.001, 0.01 and 0.1 respectively. We found that the smaller the value was, the longer the execution time was, and after over the value of 0.001, the accuracy had no substantially change. We set it to 0.001.

4.3.3 Extremely randomized trees

Parameters were consistent with the random forest.

4.3.4 Multi-class adaboost classifier

1. For boosting methods, we tested discrete boosting, gentle boosting and real boosting respectively, found that gentle boosting is the best of all.
2. Tree depth was initialized with value of 10.
3. Shrinkage factor: we set it with value of 0.1, 0.2, 0.1 and 0.4 respectively, and found that 0.1 was the best of all.
4. Maximum boosting number: we set it to the value of 10, 50, 100, 200 and 400 respectively. We found that after over the value of 100, the precision had no significant increase. So we set it to value of 100.

4.3.5 Stochastic gradient boosting

1. Maximum depth of the tree, same as in decision tree, was set it to 10.
2. Loss function type for classification problems was generally selected for cross-entropy loss.
3. Shrinkage factor was set to 0.1.
4. Subsampling percentage was used to control the sampling percentage for every single tree in the training process, we set it to 0.8.
5. Maximum lift was set to100.

4.3.6 Support vector machines

1. Type: we set it to C - Support Vector Classification and ν - support vector classification, it was found that ν - support vector classification is better than C - support.

Penalty factor for C - support vector classification was set to 1.

Penalty factor for ν -support vector classification, value range was $[0,1]$, we set it to 0.5, p was set to 0.1.

2, Kernel types are linear, radial basis, sigmoid-type function, POLY.

Total Accurac Data sets / Method		SPECTF	Ionosphere	Spam	optdigits
C - SVM	POLY	**0.770**	**0.565**	**0.618**	**0.978**
	RFB	0.898	0.749	0.772	0.562
	Sigmoid function	0.080	0.693	0.374	0.101
ν - SVM	POLY	0.748	0.733	0.629	0.934
	RFB	0.903	0.796	0.779	0.586
	Sigmoid function	0.080	0.840	0.648	0.101

Table 4: Overall accuracy of libSVM on four data sets.

From table 4, we tested non-linear support vector machines total accuracy on four data sets. We selected ν - Support Vector POLY for classification as a non-linear support vector machines kernel type.

4.3.7 Linear support vector machines

From table 5, we can see that L2R_L2LOSS_SVC for loss function got good accuracy, so we chose L2R_L2LOSS_SVC loss function as loss type of linear support vector machines.

Total accuracy Data sets / Cost type	SPECTF	Ionosphere	Spam	Optdigits
L2R L2LOSS SVC	**0.620**	**0.856**	**0.898**	**0.947**
L2R L2LOSSSVC DUAL	0.577	0.856	0.617	0.939
L2R L1LOSS SVC DUAL	0.577	0.848	0.865	0.935
MCSVM CS	0.805	0.860	0.631	0.933

Table 5: Overall accuracy of liblinear on four data sets.

4.3.8 K nearest neighbors

Nearest neighbor K was set to the value of 1, 3, 5, 7, 9, 11, 13 and 15 respectively. The accuracy was found to reduce with increasing K value, so we set it to 1.

4.3.9 Multi-class logistic classifier

According to our single test with variable regularization factor values of 1.0e-12, 1.0e-11, 1.0e-10, 1.0e-9, 1.0e-8, 1.0e-7 and 1.0, we found that the best results is the regularization factor with value of 1.0e-9. So in our comparing experiments, the regularization factor (REG_STRENGTH) was set to 1.0e-9.

Similarly, we did same experiment, most optimum results were found with the iteration range from 800 to 980. So in our comparing experiments, maximum number of iterations was set to 1000 for avoidance of losing optimum values.

4.3.10 Multilayer perceptron

1. The propagation algorithm was backward and forward propagation respectively. Apparently backward propagation algorithm achieved significantly higher accuracy.
2. Gradient weight was typically set to 0.1.
3. The momentum, front weights reflecting differences in two iterations, was typically set to 0.1.

4.3.11 Naive Bayesian Classifier

No parameters needed to be set.

4.3.12 Conditional Random Field Classifier

We only need set the classification type in this experiments.

5 Experimental results and analysis

Firstly, we need define descriptions for the symbol in tables: DecisionTree represents decision tree classifier. RandomForest represents the random forests. ExtraTrees is extremely random tree. BoostedClassifier represents the multi-class adaboost classifier. GradientBoostTree represents stochastic gradient boosting, libSVM is support vector machines. libLinear represents linear support vector machine. Knearest represents K nearest neighbor classifier, MultiClassLogistic is multi-class logistic classifier. MultiLayerPerceptron represents multilayer perceptron.NormalBayesianNet represents the naive Bayesian classifier. CRF represents Conditional Random Fields classifier

5.1 Overall accuracy on five data sets

From table 6, on SPECTF data set, stochastic gradient boosting achieved the highest overall accuracy, average precision, recall and Jaccard value, following by non-linear support vector machine, random forest, adaboost classifiers and CRF have achieved relatively good performance.

Algorithm	Overall accuracy	Average precision	Average recall	Average jaccard coefficient
DecisionTree	0.657	0.5421	0.631	0.381
RandomForest	0.759	0.5721	0.686	0.456
ExtraTrees	0.668	0.5537	0.667	0.394
BoostedClassifier	0.716	0.5847	0.754	0.440
GradientBoostTree	**1.000**	**1.000**	**1.000**	**1.000**
libSVM	0.770	0.5664	0.662	0.458
libLinear	0.620	0.5342	0.611	0.356
Knearest	0.604	0.5666	0.724	0.362
Multi-classLogistic	0.620	0.5342	0.611	0.356
MultiLayerPerceptron	0.679	0.5377	0.612	0.391
NaiveBayesianNet	0.588	0.510	0.532	0.327
CRF	0.683	0.543	0.652	0.439

Table 6: Overall accuracy on SPECTF data set.

Algorithm	Overall accuracy	Average precision	Average recall	Average jaccard coefficient
DecisionTree	0.792	0.765	0.799	0.635
RandomForest	0.828	0.807	0.854	0.691
ExtraTrees	0.888	0.862	0.897	0.780
BoostedClassifier	0.900	0.876	0.899	0.798
GradientBoostTree	**1.000**	**1.000**	**1.000**	**1.000**
libSVM	0.749	0.735	0.775	0.584
libLinear	0.856	0.867	0.787	0.694
Knearest	0.844	0.822	0.804	0.691
Multi-classLogistic	0.768	0.732	0.750	0.593
MultiLayerPerceptron	0.768	0.742	0.775	0.603
NaiveBayesianNet	0.733	0.683	0.633	0.500
CRF	0.863	0.870	0.793	0.612

Table 7: Overall accuracy on Ionosphere data set.

From table 7, on Ionoshere datasets, stochastic gradient boosting achieved the highest overall accuracy, average precision, recall and Jaccard value, following by adaboost classifier, extremely randomized trees, CRF and linear support vector machines.

Algorithm	Overall accuracy	Average precision	Average recall	Average jaccard coefficient
DecisionTree	0.905	0.905	0.896	0.819
RandomForest	0.930	0.936	0.918	0.862
ExtraTrees	0.863	0.897	0.832	0.738
BoostedClassifier	0.940	0.942	0.933	0.882
GradientBoostTree	**0.981**	**0.981**	**0.980**	**0.962**
libSVM	0.618	0.687	0.522	0.332
libLinear	0.898	0.892	0.901	0.811
Knearest	0.775	0.765	0.764	0.622
Multi-classLogistic	0.923	0.921	0.918	0.852
MultiLayerPerceptron	0.908	0.904	0.905	0.827
NaiveBayesianNet	0.602	0.801	0.500	0.301
CRF	0.784	0.778	0.790	0.653

Table 8: Overall accuracy on spam data set.

From table 8, on the spam dataset stochastic gradient boosting accuracy achieved the highest overall average precision, recall and Jaccard value, following by multi-lass adaboost classifier, random forests. For binary classification problem but, under the low-dimensional data set, the stochastic gradient boosting and ensemble classifiers such as multi-class adaboost showed a high performance.

From table 9, on optdigits data set, for the multi-classification problem, stochastic gradient boosting achieved the highest overall accuracy, average precision, recall and Jaccard value. K-nearest neighbor, non-linear support vector machine, random forests, extremely randomized trees and CRF also achieved good overall performance. Linear support vector machines, many types of logic, multi-layer perceptron is manifested in ordinal performance, and decision trees and Naïve Bayes classifier achieved the worst performance of all.

Algorithm	Overall accuracy	Average precision	Average recall	Average jaccard coefficient
DecisionTree	0.847	0.850	0.847	0.738
RandomForest	0.966	0.966	0.966	0.936
ExtraTrees	0.969	0.970	0.969	0.942
BoostedClassifier	0.870	0.880	0.870	0.778
GradientBoostTree	**1.000**	**1.000**	**1.000**	**1.000**
libSVM	0.978	0.979	0.978	0.959
libLinear	0.947	0.948	0.946	0.901
Knearest	0.979	0.980	0.979	0.961
Multi-classLogistic	0.943	0.944	0.943	0.893
MultiLayerPerceptron	0.946	0.947	0.946	0.897
NaiveBayesianNet	0.843	0.881	0.844	0.732
CRF	0.960	0.962	0.958	0.932

Table 9: Overall accuracy on optdigits data set.

From table 10, due to the high dimensional feature data of Scene15 data set, multi-layer perceptron, naive Bayesian classifier and nonlinear support vector machine were failed. This mainly caused by the intrinsic shortcomes of nonlinear SVM, Bayesian networks and multi-layer perceptron. They have too many inner loops and intermediate phased which need computation and storage. This situation is worse when feature data is high dimensional. Another reason is that our testing environment is limited. If we have enough memory and strong CPU capability, I think this phenomenon will disappear. In another view, this also reveals that they

Algorithm	Overall accuracy	Average precision	Average recall	Average Jaccard coefficient
DecisionTree	0.397	0.391	0.376	0.246
RandomForest	0.531	0.548	0.493	0.339
ExtraTrees	0.655	0.639	0.637	0.479
BoostedClassifier	0.422	0.502	0.424	0.274
GradientBoostTree	**0.999**	**0.999**	**0.999**	**0.999**
libSVM	Invalid	Invalid	Invalid	Invalid
libLinear	0.815	0.813	0.8113	0.694
Knearest	0.565	0.602	0.5421	0.382
Multi-classLogistic	0.794	0.797	0.7915	0.665
MultiLayerPerceptron	Invalid	Invalid	Invalid	Invalid
NaiveBayesianNet	Invalid	Invalid	Invalid	Invalid
CRF	0.650	0.625	0.621	0.456

Table 10: Overall accuracy on Scene15 data set.

Algorithm	F_1		Jaccard coefficient	
	Class0	Class1	Class0	Class1
RandomForest	0.286	0.855	0.167	0.747
ExtraTrees	0.244	0.788	0.139	0.650
BoostedClassifier	0.312	0.822	0.185	0.697
GradientBoostTree	**1.000**	**1.000**	**1.000**	**1.000**
libSVM	0.271	0.863	0.157	0.760
libLinear	0.202	0.751	0.113	0.601
Knearest	0.260	0.730	0.149	0.575
Multi-classLogistic	0.202	0.751	0.113	0.601
MultiLayerPerceptron	0.211	0.799	0.118	0.665
NaiveBayesianNet	0.154	0.728	0.083	0.572
CRF	0.278	0.813	0.156	0.745

Table 11: Inter-class accuracy on SPECTF data set.

Algorithm	F_1		Jaccard coefficient	
	Class0	Class1	Class0	Class1
RandomForest	0.919	0.840	0.850	0.724
ExtraTrees	0.916	0.835	0.844	0.717
BoostedClassifier	0.926	0.847	0.863	0.734
GradientBoostTree	**1.000**	**1.000**	**1.000**	**1.000**
libSVM	0.796	0.674	0.661	0.508
libLinear	0.903	0.723	0.824	0.566
Knearest	0.890	0.735	0.802	0.581
Multi-classLogistic	0.827	0.651	0.706	0.482
MultiLayerPerceptron	0.820	0.678	0.695	0.513
NaiveBayesianNet	0.822	0.464	0.698	0.302
CRF	0.908	0.838	0.846	0.721

Table 12: Inter-class accuracy on Ionosphere data set.

Algorithm	F_1		Jaccard coefficient	
	Class0	Class1	Class0	Class1
RandomForest	0.944	0.908	0.894	0.831
ExtraTrees	0.897	0.798	0.814	0.664
BoostedClassifier	0.952	0.923	0.908	0.857
GradientBoostTree	**0.985**	**0.977**	**0.970**	**0.955**
libSVM	0.757	0.107	0.609	0.056
libLinear	0.914	0.878	0.841	0.783
Knearest	0.815	0.715	0.687	0.557
Multi-classLogistic	0.937	0.903	0.881	0.823
MultiLayerPerceptron	0.924	0.886	0.859	0.795
NaiveBayesianNet	0.752	0.002	0.603	0.001
CRF	0.933	0.891	0.885	0.838

Table 13: Inter-class accuracy statistics on Spam data set.

have tough condition for real application.

However, stochastic gradient boosting achieved the highest overall accuracy, average precision, recall and Jaccard value, following by linear support vector machines, multi-class logistic regression. Other algorithms got poor performance.

5.2 Inter class accuracy and jaccard coefficient evaluation on five data sets

The statistical results above reflected the overall performance of the algorithms. However, inter classes F_1 and Jaccard could reflect more detailed information. Accuracy of each class may vary greatly due to differences of the data. Overall each accuracy was determined by average of sum each class accuracy.

From table 11, on the data set SPECTF, class 0 represents normal, class 1 represents abnormal. Stochastic gradient boosting was fully recognized, so it had the highest F_1 and Jaccard coefficients in each sub class (both class 0 and class 1). Remains of algorithms' F_1 and Jaccard coefficients were not high in class 0, however there were high accuracy in class 1. This

Algorithm		Class0	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8	Class9
DecisionTree	F_1	0.933	0.830	0.813	0.817	0.803	0.891	0.940	0.836	0.777	0.832
	Jaccard	0.875	0.710	0.685	0.690	0.671	0.804	0.887	0.719	0.636	0.712
RandomForest	F_1	0.989	0.965	0.983	0.962	0.981	0.975	0.986	0.972	0.920	0.933
	Jaccard	0.978	0.933	0.966	0.926	0.962	0.952	0.973	0.945	0.851	0.875
ExtraTrees	F_1	0.992	0.963	0.994	0.956	0.986	0.975	0.989	0.972	0.938	0.934
	Jaccard	0.983	0.928	0.989	0.915	0.973	0.952	0.978	0.945	0.883	0.877
BoostedClassifier	F_1	0.960	0.804	0.930	0.813	0.899	0.919	0.960	0.850	0.783	0.798
	Jaccard	0.923	0.672	0.870	0.685	0.816	0.850	0.924	0.740	0.644	0.664
GradientBoostTree	F_1	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	Jaccard	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
libSVM	F_1	0.997	0.978	0.994	0.972	0.994	0.986	0.997	0.977	0.948	0.943
	Jaccard	0.994	0.957	0.989	0.946	0.989	0.973	0.995	0.956	0.901	0.892
libLinear	F_1	0.992	0.914	0.983	0.942	0.964	0.941	0.983	0.954	0.890	0.911
	Jaccard	0.983	0.842	0.966	0.889	0.930	0.889	0.967	0.912	0.801	0.837
Knearest	F_1	1.000	0.965	0.994	0.978	0.981	0.986	1.000	0.989	0.956	0.949
	Jaccard	1.000	0.933	0.989	0.957	0.962	0.973	1.000	0.978	0.916	0.904
Multi-classLogistic	F_1	0.977	0.957	0.963	0.934	0.961	0.937	0.978	0.946	0.892	0.885
	Jaccard	0.956	0.918	0.929	0.877	0.925	0.881	0.957	0.897	0.805	0.794
MultiLayerPerceptron	F_1	0.986	0.948	0.938	0.925	0.951	0.957	0.962	0.949	0.913	0.930
	Jaccard	0.973	0.902	0.884	0.861	0.906	0.918	0.926	0.902	0.840	0.869
NaiveBayesianNet	F_1	0.942	0.914	0.940	0.891	0.011	0.967	0.949	0.762	0.793	0.883
	Jaccard	0.890	0.842	0.886	0.804	0.006	0.936	0.902	0.616	0.656	0.791
CRF											

Table 14: Inter-class accuracy on optdigits data set.

indicated that the overall accuracy was boosted by the accuracy of the class 1. Distribution trends of Jaccard coefficient was in accordance with that of F_1. This meant that the higher F_1 was, the higher Jaccard coefficient was in class 1. Further, support vector machines, random forests and CRF also had high F_1 and Jaccard coefficient in class 1.

From table 12, on data set Ionosphere, class 0 represents the presence of the fact, class 1 represents no presence of the fact. Performance was higher in class 0 than that in class 1. Stochastic gradient boosting to achieve the highest value. Random forests, CRF and multi-class adaboost classifier also got good performance.

From table 14, on optdigits dataset, class 0 to class 9 represent ten digit of 0 to 9. All algorithms had relatively equal performance in each class. Stochastic gradient boosting, the non-linear support vector machines, random forests, Multilayer Perceptron, linear multi-class support vector machines and multi-class logistic classifier had high performance in each class.

Algorithm		Class0	Class1	Class2	Class3	Class4	Class5	Class6	Class7
DecisionTree	F_1	0.238	0.618	0.718	0.550	0.304	0.397	0.471	0.397
	Jaccard	0.135	0.447	0.561	0.380	0.179	0.247	0.308	0.248
RandomForest	F_1	0.848	0.654	0.852	0.327	0.491	0.571	0.618	0.688
	Jaccard	0.736	0.486	0.742	0.196	0.326	0.400	0.448	0.524
ExtraTrees	F_1	0.805	0.816	0.859	0.779	0.590	0.744	0.652	0.777
	Jaccard	0.673	0.689	0.752	0.638	0.419	0.593	0.483	0.635
BoostedClassifier	F_1	0.606	0.126	0.290	0.415	0.531	0.399	0.447	0.608
	Jaccard	0.435	0.067	0.170	0.262	0.362	0.249	0.287	0.437
GradientBoostTree	F_1	**1.000**	**1.000**	**1.000**	**1.000**	**0.997**	**1.000**	**1.000**	**1.000**
	Jaccard	**1.000**	**1.000**	**1.000**	**1.000**	**0.994**	**1.000**	**1.000**	**1.000**
libSVM	F_1	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
	Jaccard	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
libLinear	F_1	0.959	0.860	0.931	0.864	0.784	0.887	0.823	0.890
	Jaccard	0.921	0.754	0.871	0.761	0.645	0.797	0.699	0.803
Knearest	F_1	0.606	0.126	0.290	0.415	0.531	0.399	0.447	0.608
	Jaccard	0.435	0.067	0.170	0.262	0.362	0.249	0.287	0.437
Multi-classLogistic	F_1	0.941	0.801	0.873	0.820	0.736	0.866	0.770	0.871
	Jaccard	0.936	0.701	0.827	0.721	0.617	0.712	0.656	0.776
MultiLayerPerceptron	F_1	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
	Jaccard	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
NaiveBayesianNet	F_1	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
	Jaccard	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
CRF	F_1	0.835	0.647	0.843	0.318	0.485	0.565	0.546	0.659
	Jaccard	0.736	0.477	0.729	0.187	0.319	0.389	0.432	0.510

Table 15: Inter-class accuracy on Scene15 data set.

Algorithm		Class8	Class9	Class10	Class11	Class12	Class12	Class14
DecisionTree	F_1	0.463	0.330	0.142	0.234	0.220	0.326	0.280
	Jaccard	0.301	0.198	0.076	0.133	0.123	0.195	0.162
RandomForest	F_1	0.495	0.410	0.034	0.197	0.071	0.282	0.482
	Jaccard	0.328	0.258	0.017	0.109	0.037	0.164	0.318
ExtraTrees	F_1	0.681	0.645	0.339	0.286	0.409	0.496	0.531
	Jaccard	0.517	0.477	0.204	0.167	0.257	0.330	0.362
BoostedClassifier	F_1	0.611	0.447	0.342	0.320	0.325	0.356	0.485
	Jaccard	0.440	0.288	0.206	0.190	0.194	0.216	0.320
GradientBoostTree	F_1	**0.997**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	Jaccard	**0.994**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
BoostedClassifier	F_1	0.623	0.459	0.309	0.288	0.324	0.415	0.528
	Jaccard	0.452	0.298	0.183	0.168	0.193	0.262	0.359
libSVM	F_1	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
	Jaccard	0.000	0.000	0.000	0.000	0.000	0.000	0.000
libLinear	F_1	0.862	0.898	0.633	0.645	0.701	0.663	0.685
	Jaccard	0.758	0.815	0.463	0.476	0.539	0.495	0.521
Knearest	F_1	0.611	0.447	0.342	0.320	0.325	0.356	0.485
	Jaccard	0.440	0.288	0.206	0.190	0.194	0.216	0.320
Multi-classLogistic	F_1	0.847	0.898	0.721	0.613	0.761	0.721	0.655
	Jaccard	0.889	0.668	0.775	0.694	0.582	0.764	0.625
MultiLayerPerceptron	. F_1 .	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
	Jaccard	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
NaiveBayesianNet	F_1	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
	Jaccard	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid	Invalid
CRF	F_1	0.488	0.402	0.028	0.188	0.066	0.277	0.466
	Jaccard	0.313	0.248	0.016	0.100	0.036	0.155	0.309

Table 16: Inter-class accuracy on Scene15 data set (cont.)

Algorithm	Binary data sets			Multi-class data sets	
	SPECTF	Ionosphere	Spam	optdigits	Scene 15
DecisionTree	1	30.9	79.9	202.9	19.562(second)
RandomForest	93.9	326.9	2.819	8112	34(minute)
ExtraTrees	266.0	437.0	3.036	12541	1hour22 minute
BoostedClassifier	202.9	451.9	1761	2.697	477.873（second）
GradientBoostTree	437.0	656.0	6732	53807	8hour42 minute
libSVM	46.9	108.9	915184	3276	Invalid
libLinear	389.9	749.0	3.165	1006	2hour10 minute
Knearest	1	16.0	280.9	708.9	120.054（second）
Multi-classLogistic	30.9	46.9	377.9	2153	190.660（second）
MultiLayerPerceptron	**857.9**	**4306**	**33727**	**172847**	**Invalid**
NaiveBayesianNet	30.9	93.0	63.0	375.9	Invalid
CRF	168.9	235.2	1.522	6420	66 minute

Table 17: Running time of twelve algorithms on five data sets (unit: millisecond except scene 15 data set).

From table 15 and table 16, on Scene15, class 0 to class 14 represented fifteen classes. Multilayer perceptron and non-support vector machines were failed because of computation cost, and naive Bayesian classifier was failed due to the huge storage. Stochastic gradient boosting, linear support vector machines achieved good performance, following by multi-class logistic classifier.

5.3 Running time performance on five data sets

From table 17, the running time of the 11 kinds by an algorithm on the five data sets can be seen that:

1. On a small data sets (SPECTE and Ionosphere), running time of multilayer perceptron was significantly slower than that of other algorithms, while other algorithms' running time were almost same. Linear support vector machines' (based on liblinear) running time was inversely lower than that of nonlinear support vector machines based on libsvm.
2. On the large data sets (spam, optidigits, scene15), the differences of running time were significant. It is clear that the linear support vector machines were significantly faster than the non-linear support vector machines.
3. For tree classifiers, decision tree was the fastest of all, following by random forests. The slowest was extremely randomized trees.
4. For boosting methods, stochastic gradient boosting was slower than the multiclass adaboost.
5. Due to the large dimensionality of data, non-linear support vector machines, and Bayesian multi-layer perceptron did not succeed.
6. CRF running time is better than ExtraTrees, but slower than RandomForest.
7. In short, for running time efficiency, naive Bayes classifiers, K nearest neighbor and decision tree were basically fast, following by random forests, multi-class logic and linear support vector machines. The stochastic gradient boosting was the slowest of all.

6 Conclusion and future work

This article compares 12 kinds of commonly used multi-classification algorithm. In the experiments, we found that:

1. The same algorithm on different data sets showed different performance. It was the key to choose a more adaptive algorithm based on the data set.
2. Stochastic gradient boosting achieved the best classification accuracy in all test data sets, but its running time was slower than other algorithms except the multilayer perceptron.
3. The composite classifiers performed well than single classifier. For example, stochastic gradient boosting, random forest, extremely randomized trees were all better than the basic decision tree. However at the same time, the more complex combination model was, the longer running time was.
4. Linear support vector machines achieved good results of both accuracy and total execution on large data sets while compared with the nonlinear support vector machine.

There are still some deficiencies in our comparative study, further research is need:

1. We compare only the basic algorithm of 12 kinds of algorithms, every algorithm has its variants, which are better than the original algorithms.
2. How to choose the optimal parameter settings for algorithm is critical for its performance. There are still more works need to be done.
3. When we deal with large data sets, how much sample should choose for training? How to find the best balance between training time and accuracy is worthy of further exploration.
4. The combination of classifiers can often lead to higher accuracy, but as mentioned above, model training time will significantly increase. Stochastic gradient boosting were obtained good accuracy in our tests on five data sets, however the running time is longer. Actually stochastic gradient boosting have many sub routine which has sub-iteration, so this will elapse many running time. Because the main routine are highly correlate the sub-iteration, so it cannot directly parallel the sub-iteration.

How to improve the running time performance with a little bit of decrease in accuracy is a meaningful research, in other words, we need to find a balance between accuracy and running time performance.

Acknowledgement

This work was supported in part by Scientific Research

Fund of Sichuan Provincial Education Department of China (No. 15TD0038) and Fundamental Research Funds for the Central University.

References

[1] ZHU J, ROSSET S, ZOU H, et al. Multi-class adaboost [J]. Ann Arbor, 2006, 1001(48109): 1612.

[2] SELFRIDGE O G. Pandemonium: a paradigm for learning in mechanization of thought processes [J]. 1958,

[3] KANAL L. Patterns in pattern recognition: 1968-1974 [J]. Information Theory, IEEE Transactions on, 1974, 20(6): 697-722.

[4] MINSKY M L. Logical versus analogical or symbolic versus connectionist or neat versus scruffy [J]. AI magazine, 1991, 12(2): 34.

[5] ROSENBLATT F: DTIC Document, 1961.

[6] BREIMAN L. Random forests [J]. Machine learning, 2001, 45(1): 5-32.

[7] FRIEDMAN J, HASTIE T, TIBSHIRANI R. Additive logistic regression: a statistical view of boosting (With discussion and a rejoinder by the authors) [J]. The annals of statistics, 2000, 28(2): 337-407.

[8] FRENCH S. Group consensus probability distributions: A critical survey [J]. Bayesian statistics, 1985, 2(183-202.

[9] BENEDIKTSSON J A, SWAIN P H. Consensus theoretic classification methods [J]. Systems, Man and Cybernetics, IEEE Transactions on, 1992, 22(4): 688-704.

[10] BERNARDO J M, SMITH A F. Bayesian theory [J]. Measurement Science and Technology, 2001, 12(2): 221.

[11] FAN R E, CHANG K W, HSIEH C J, et al. LIBLINEAR: A library for large linear classification [J]. The Journal of Machine Learning Research, 2008, 9(1871-4.

[12] DUIN R, TAX D. Experiments with classifier combining rules [J]. Multiple classifier systems, 2000, 16-29.

[13] ALY M. Survey on multi-class classification methods [J]. Neural networks, 2005, 1-9.

[14] NILLSON N. Learning machines: Foundations of trainable pattern classifying systems [M]. McGraw-Hill, New York. 1965.

[15] KING R D, FENG C, SUTHERLAND A. Statlog: comparison of classification algorithms on large real-world problems [J]. Applied Artificial Intelligence an International Journal, 1995, 9(3): 289-333.

[16] BAUER E, KOHAVI R. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants [J]. Machine learning, 1998, 36(1): 2.

[17] LECUN Y, JACKEL L, BOTTOU L, et al. Comparison of learning algorithms for handwritten digit recognition; proceedings of the International conference on artificial neural networks, F, 1995 [C].

[18] DING C H, DUBCHAK I. Multi-class protein fold recognition using support vector machines and neural networks [J]. Bioinformatics, 2001, 17(2): 107-38.

[19] LI T, ZHANG C, OGIHARA M. A comparative study of feature selection and multi-class classification methods for tissue classification based on gene expression [J]. Bioinformatics, 2004, 20(15): 2429-37.

[20] FOODY G M, MATHUR A. A relative evaluation of multi-class image classification by support vector machines [J]. Geoscience and Remote Sensing, IEEE Transactions on, 2004, 42(6): 1335-43.

[21] HSU C-W, LIN C-J. A comparison of methods for multi-class support vector machines [J]. Neural Networks, IEEE Transactions on, 2002, 13(2): 415-25.

[22] CARUANA R, NICULESCU-MIZIL A. An empirical comparison of supervised learning algorithms; proceedings of the Proceedings of the 23rd international conference on Machine learning, F, 2006 [C]. ACM.

[23] KRUSIENSKI D J, SELLERS E W, CABESTAING F, et al. A comparison of classification techniques for the P300 Speller [J]. Journal of neural engineering, 2006, 3(4): 299.

[24] WITTEN I H, FRANK E. Data Mining: Practical machine learning tools and techniques [M]. Morgan Kaufmann, 2005.

[25] MURPHY K P. Machine Learning: a Probabilistic Perspective [J]. 2012,

[26] QUINLAN J R. Induction of decision trees [J]. Machine learning, 1986, 1(1): 81-106.

[27] BREIMAN L, FRIEDMAN J, STONE C J, et al. Classification and regression trees [M]. Chapman & Hall/CRC, 1984.

[28] HO T K. Random decision forests; proceedings of the Document Analysis and Recognition, 1995, Proceedings of the Third International Conference on, F, 1995 [C]. IEEE.

[29] BREIMAN L. Bagging predictors [J]. Machine learning, 1996, 24(2): 123-40.

[30] BREIMAN L. Statistical modeling: The two cultures (with comments and a rejoinder by the author) [J]. Statistical Science, 2001, 16(3): 199-231.

[31] IVERSON L R, PRASAD A M, MATTHEWS S N, et al. Estimating potential habitat for 134 eastern US tree species under six climate scenarios [J]. Forest Ecology and Management, 2008, 254(3): 390-406.

[32] GEURTS P, ERNST D, WEHENKEL L. Extremely randomized trees [J]. Machine learning, 2006, 63(1): 3-42.

[33] HASTIE T, TIBSHIRANI R, FRIEDMAN J, et al. The elements of statistical learning: data mining, inference and prediction [J]. The Mathematical Intelligencer, 2005, 27(2): 83-5.

[34] FREUND Y, SCHAPIRE R E. A decision-theoretic generalization of on-line learning and an application

to boosting [J]. Journal of computer and system sciences, 1997, 55(1): 119-39.

[35] FRIEDMAN J H. Stochastic gradient boosting [J]. Computational Statistics & Data Analysis, 2002, 38(4): 367-78.

[36] BURGES C J C. A tutorial on support vector machines for pattern recognition [J]. Data mining and knowledge discovery, 1998, 2(2): 121-67.

[37] CORTES C, VAPNIK V. Support-vector networks [J]. Machine learning, 1995, 20(3): 273-97.

[38] KEERTHI S S, SUNDARARAJAN S, CHANG K-W, et al. A sequential dual method for large scale multi-class linear SVMs; proceedings of the Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, F, 2008 [C]. ACM.

[39] CRAMMER K, SINGER Y. On the algorithmic implementation of multi-class kernel-based vector machines [J]. The Journal of Machine Learning Research, 2002, 2(265-92.

[40] COVER T, HART P. Nearest neighbor pattern classification [J]. Information Theory, IEEE Transactions on, 1967, 13(1): 21-7.

[41] LIU D C, NOCEDAL J. On the limited memory BFGS method for large scale optimization [J]. Mathematical programming, 1989, 45(1): 503-28.

[42] RONSENBLATT F. The perceptron: a probabilistic model for information storage and organization in the brain [J]. Psychological review, 1958, 65(386-408.

[43] BISHOP C M. Pattern recognition and machine learning [M]. Springer New York, 2006.

[44] WASSERMAN P D, SCHWARTZ T. Neural networks. II. What are they and why is everybody so interested in them now? [J]. IEEE Expert, 1988, 3(1): 10-5.

[45] Hang Li. Statistic Learning [M]. Tsinghua press, 2012.

[46] COLLOBERT R, BENGIO S. Links between Perceptron, MLPs and SVMs; proceedings of the Proceedings of the twenty-first international conference on Machine learning, F, 2004 [C]. ACM.

[47] GOULD S. DARWIN: A Framework for Machine Learning and Computer Vision Research and Development [J]. Journal of Machine Learning Research, 2012, 13(12): 3499-503.

[48] CHANG C C, LIN C J. LIBSVM: a library for support vector machines [J]. ACM Transactions on Intelligent Systems and Technology (TIST), 2011, 2(3): 27.

[49] FAWCETT T. An introduction to ROC analysis [J]. Pattern recognition letters, 2006, 27(8): 861-74.

[50] DAVIS J, GOADRICH M. The relationship between Precision-Recall and ROC curves; proceedings of the Proceedings of the 23rd international conference on Machine learning, F, 2006 [C]. ACM.

[51] JACCARD P. The distribution of the flora in the alpine zone. 1 [J]. New Phytologist, 2006, 11(2): 37-50.

[52] ASUNCION A, NEWMAN D J. UCI machine learning repository [M]. 2007.

[53] KURGAN L A, CIOS K J, TADEUSIEWICZ R, et al. Knowledge discovery approach to automated cardiac SPECT diagnosis [J]. Artificial Intelligence in Medicine, 2001, 23(2): 149-69.

[54] LAZEBNIK S, SCHMID C, PONCE J. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories; proceedings of the Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, F, 2006 [C]. IEEE.

[55] VEDALDI A, ZISSERMAN A. Efficient additive kernels via explicit feature maps [J]. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2012, 34(3): 480-92.

[56] Bradski G, Kaehler A (2008) Learning OpenCV: Computer vision with the OpenCV library. O'Reilly Media, Incorporated

[57] Delashmit WH, Manry MT (2005) Recent developments in multilayer perceptron neural networks. In: Proceedings of the seventh Annual Memphis Area Engineering and Science Conference, MAESC.

An Efficient Algorithm for Mining Frequent Closed Itemsets

Gang Fang [1, 2], Yue Wu [1], Ming Li [1] and Jia Chen [1]
[1]School of Computer Science and Engineering, University of Electronic Science and Technology of China,
Chengdu, Sichuan, 611731, P. R. China
[2]School of Computer Science and Engineering, Chongqing Three Gorges University,
Wanzhou, Chongqing, 404000, P. R. China
E-mail: gangfang07@sohu.com, ywu@uestc.edu.cn, ming.m.li@alcatel-lucent.com, jchen@uestc.edu.cn

Keywords: frequent closed itemsets, Galois connection, granular computing, association rules, data mining

Received: February 6, 2014

To avoid generating an undesirably large set of frequent itemsets for discovering all high confidence association rules, the problem of finding frequent closed itemsets in a formal mining context is proposed. In this paper, aiming to these shortcomings of typical algorithms for mining frequent closed itemsets, such as the algorithm A-close and CLOSET, we propose an efficient algorithm for mining frequent closed itemsets, which is based on Galois connection and granular computing. Firstly, we present the smallest frequent closed itemsets and its characters, contain some properties and theorems, then propose a novel notion, called the smallest frequent closed granule, which can help the algorithm save reading the database to reduce the costed I/O for discovering frequent closed itemsets. And then we propose a novel model for mining frequent closed itemsets based on the smallest frequent closed granules, and a connection function for generating the smallest frequent closed itemsets. The generator function create the power set of the smallest frequent closed itemsets in the enlarged frequent 1-item manner, which can efficiently avoid generating an undesirably large set of candidate smallest frequent closed itemsets to reduce the costed CPU and the occupied main memory for generating the smallest frequent closed granules. Finally, we describe the algorithm for the proposed model. On these different datasets, we report the performances of the algorithm and its trend of the performances to discover frequent closed itemsets, and further discuss how to solve the bottleneck of the algorithm. For mining frequent closed itemsets, all these experimental results indicate that the performances of the algorithm are better than the traditional and typical algorithms, and it also has a good scalability. It is suitable for mining dynamic transactions datasets.

Povzetek: Opisan je nov algoritem asociativnega učenja za pogoste entitete.

1 Introduction

Association rules mining is introduced in [1], Agrawal et al. firstly propose a classic algorithm for discovering association rules in [2], namely, the Apriori algorithm. However, it is also well known that mining frequent patterns often generates a very large number of frequent itemsets and association rules, which reduces not only efficiency but also effectiveness of mining since users have to sift through a large number of mined rules to discover useful ones. In order to avoid the shortcoming, Pasquier et al. introduce the problems of mining frequent closed itemsets in [3], and propose an efficient Apriori-based mining algorithm, called A-close. Subsequent, Zaki and Hsiao propose another mining algorithm in [4], called CHARM, which improves mining efficiency by exploring an item-based data structure. However, we find A-close and CHARM are still costly when mining long patterns or low minimum support thresholds in large database, especially, CHARM depends on the given data structure and need the overlarge memory. As a continued study on frequent patterns mining without candidate generation in [5], J. Pei et al. propose an efficient method for mining frequent closed itemsets without candidate generation in [6], called CLOSET. There are more study works for mining frequent closed itemsets in [7-13]. The familiar algorithms include MAFIA in [7], CLOSE+ in [8] and DCI-CLOSED in [9].

At present, for mining frequent closed itemsets, there are two types of main current methods as follows:

The first is the method of mining frequent closed itemsets with candidate based on the Apriori algorithm in [3 and 14]. The A-close algorithm in [3] is a well-known typical algorithm for the first method, which adopts the bottom-up search strategy as the Apriori-like in [2], and constructs the set of generators in a level-wise manner: $(i+1)-generators$ are created by joining $i-generators$. For the first method, the advantages are the less usage of memory, simple data structure, and easy implementing it and maintaining; its disadvantages are the more occupied CPU for matching candidate patterns, and the overlarge costed I/O for the repeatedly scanning the database to compute the support.

The second is the method of mining frequent closed itemsets without candidate based on the FP-tree structure in [6, 15 and 16]. The CLOSET algorithm in [6] is an extended study of the FP-Growth for mining frequent patterns in [5]. For the second method, the advantages are reducing the overlarge computing corresponding to the joined potential generators in the A-close algorithm, and saving the costed I/O of reading the database. But it has these disadvantages, such as complex data structure costs more memory, creating recursion FP-tree occupies more CPU, and implementing it is troublesome.

Rough set theory in [17] and formal concept analysis in [18 and 19] are two efficient methods for the representation and discovery of knowledge in [20 and 21]. Rough set theory and formal concept analysis are actually related and often complementary approaches to data analysis, but rough set models enable us to precisely define and analyse many notions of granular computing in [22 and 23].

Reference [22] develops a general framework for the study of granular computing and knowledge reduction in formal concept analysis. In formal concept analysis, granulation of the universe of discourse, description of granules, relationship between granules, and computing with granules are issues that need further scrutiny. Since the basic structure of a concept lattice induced from a formal context is the set of object concepts and every formal concept in the concept lattice can be represented as a join of some object concepts, each object concept can be viewed as an information granule in the concept lattice.

An important notion in formal concept analysis is thus a formal concept, which is a pair consisting of a set of objects (the extension) and a set of attributes (the intension) such that the intension consists of exactly those attributes that the objects in the extension have in common, and the extension contains exactly those objects that share all attributes in the intension in [22]. For the study of granular computing, the formal concept is defined as a granule, such as an information granule.

Based on the notions of granularity in [24] and abstraction in [25], the ideas of granular computing have been widely investigated in artificial intelligence in [26], such as, granular computing has been applied to association rules mining in [27 and 28], where a partition model of granular computing is applied to constructing information granule in [26], which depends on rough set theory in [29] and quotient space theory in [30].

In this paper, we propose a novel model based on granular computing, namely, an efficient algorithm for mining frequent closed itemsets, which constructs the set of generators in the enlarged frequent $1-item$ manner to reduce the costed CPU, and adopts granular computing to reduce the costed I/O.

The rest of the paper is organized as follows:

In Section 2, we present the related concepts with closed itemset and granular computing; In Section 3, we propose a novel model for mining frequent closed itemsets based on granular computing; In Section 4, we describe the efficient mining algorithm; Section 5 reports the performance comparison of our with A-close and CLOSET. In Section 6, we summarize study work and discuss some future research directions.

2 Related concepts

In this section, referring to the definitions and theorems in [3, 4, 6, and 22], we present the following *definitions*, *properties*, *theorems*, and *propositions* with closed itemsets and granular computing.

Definition 2.1 A formal context is a triplet $D = (U, A, R)$, where

$U = \{u_1, u_2, ..., u_n\}$ $(n = \square U \;)\square$, called the universe of discourse, is a finite nonempty set of objects;

$A = \{a_1, a_2, ..., a_m\}$ $(m = \square A \;)\square$, called the attributes set, is also a finite nonempty set of attributes;

$R \subseteq U \times A$, called the relations, is a binary relation between objects U and attributes A, where each couple $(u, a) \in R$ denotes the fact that the object u $(u \in U)$ is related to the attribute a $(a \in A)$.

Here, we make the following ratiocinations become concise, and then let the attribute $a(a \in A)$ be Boolean, where each attribute is regarded as an item, i.e. the attributes set A is a general itemset. In fact, these ratiocinations are also suitable for the quantitative attributes.

Definition 2.2 Galois connection, let $D = (U, A, R)$ be a formal context, for $O \subseteq U$ and $I \subseteq A$, we define:

$\mu(O) : P(U) \to P(A)$, namely

$\mu(O) = \{i \in A \mid \forall o \in O, (o, i) \in R\}$, which denotes the maximal set of items shared by all objects o $(o \in O)$;

$\ddot{}\,(I) : P(A) \to P(U)$, namely

$\ddot{}\,(I) = \{o \in U \mid \forall i \in I, (o, i) \in R\}$, which denotes the maximal set of objects that have all items i $(i \in I)$;

And the couple of applications $(\mu, \ddot{}\,)$ is defined as a Galois connection between the power set of U (i.e. $P(U)$) and the power set of A (i.e. $P(A)$).

Property 2.1 For a formal context $D = (U, A, R)$, if $O, O_1, O_2 \subseteq U$ and $I, I_1, I_2 \subseteq A$, then we have:

(1) $I_1 \subseteq I_2 \Rightarrow \ddot{}\,(I_1) \supseteq \ddot{}\,(I_2)$;

(1*) $O_1 \subseteq O_2 \Rightarrow \mu(O_1) \supseteq \mu(O_2)$;

(2) $I \subseteq \mu(O) \Leftrightarrow O \subseteq \ddot{}\,(I)$.

Definition 2.3 Galois closure operators are defined as the operators $h = \mu \circ \ddot{}$ in $P(A)$ and $\hbar = \ddot{}\, \circ \mu$ in $P(U)$, where they are also expressed as the following notation: $h(I) = \mu \circ \ddot{}\,(I) = \mu(\ddot{}\,(I)), \hbar(O) = \ddot{}\, \circ \mu(O) = \ddot{}\,(\mu(O))$.

Property 2.2 For a formal context $D = (U, A, R)$, let $(\mu, \ddot{}\,)$ be the Galois connection. If $O, O_1, O_2 \subseteq U$ and I, $I_1, I_2 \subseteq A$, then we have:

Extension: (3) $I \subseteq h(I)$; (3*) $O \subseteq \hbar(O)$;

Idempotency: (4) $h(h(I)) = h(I)$;

(4*) $\hbar(\hbar(O)) = \hbar(O)$;

Monotonicity: (5) $I_1 \subseteq I_2 \Rightarrow h(I_1) \subseteq h(I_2)$;

(5*) $O_1 \subseteq O_2 \Rightarrow \hbar(O_1) \subseteq \hbar(O_2)$;

Definition 2.4 Closed itemsets, an itemsets $C \subseteq A$ from D is a closed itemset if and only if $h(C) = C$. The smallest (minimal) closed itemset containing an itemset I is obtained by applying h to I.

Here, we call $h(I)$ the closure of I.

Theorem 2.1 For a formal context $D = (U, A, R)$, let $I_1, I_2 \subseteq A$ be two itemsets. We have:

$h(I_1 \cup I_2) = h(h(I_1) \cup h(I_2))$.

Proof. Let $I_1, I_2 \subseteq A$ be two itemsets.

$\because I_1 \subseteq h(I_1), I_2 \subseteq h(I_2)$ (Extension)

$\therefore I_1 \cup I_2 \subseteq h(I_1) \cup h(I_2)$

$\therefore h(I_1 \cup I_2) \subseteq h(h(I_1) \cup h(I_2))$ (Monotonicity)

And $\because I_1 \subseteq I_1 \cup I_2, I_2 \subseteq I_1 \cup I_2$

$\therefore h(I_1) \subseteq h(I_1 \cup I_2), h(I_2) \subseteq h(I_1 \cup I_2)$

$\therefore h(h(I_1) \cup h(I_2)) \subseteq h(h(I_1 \cup I_2))$ (Monotonicity)

$\therefore h(h(I_1) \cup h(I_2)) \subseteq h(I_1 \cup I_2)$ (Idempotency)

$\therefore h(I_1 \cup I_2) = h(h(I_1) \cup h(I_2))$.

Proposition 2.1 For a formal context $D = (U, A, R)$, the closed itemset $h(I)$ corresponding to the closure by h of the itemset $I (I \subseteq A)$ is the intersection of all objects in U that contain I:

$h(I) = \bigcap_{o \in U} \{\mu(\{o\}) \mid I \subseteq \mu(\{o\})\}$.

Proof. Let $H = \bigcap_{o \in S} \mu(\{o\})$, where

$S = \{o \in U \mid I \subseteq \mu(\{o\})\}$. And we have

$h(I) = \mu(\text{¨}(I)) = \bigcap_{o \in \text{¨}(I)} \mu(\{o\}) = \bigcap_{o \in S^\circ} \mu(\{o\})$, where

$S^\circ = \{o \in U \mid o \in \text{¨}(I)\}$.

Let's show that $S^\circ = S$, i.e. $I \subseteq \mu(\{o\}) \Leftrightarrow o \in \text{¨}(I)$.

$\because \text{¨}(I) \supseteq \{o\}; \quad \therefore \mu(\text{¨}(I)) \subseteq \mu(\{o\})$ (Property 2.1)

$\because I \subseteq \mu(\text{¨}(I))$ (Extension)

$\therefore o \in \text{¨}(I) \Leftrightarrow I \subseteq \mu(\text{¨}(I)) \subseteq \mu(\{o\})$

We have $S = S^\circ$, and also have $h(I) = H$.

Definition 2.5 Formal granule, for a formal context $D = (U, A, R)$, a two-tuple $G = < I, \text{¨}(I) >$ is defined as a formal granule of the context $D = (U, A, R)$, where

I, called the intension of formal granule, is an abstract description of common features or properties shared by objects in the extension, which is expressed as $I = \{i_1, i_2, ..., i_k\} (I \subseteq A, k = □ I □)$.

$\text{¨}(I)$, called the extension of formal granule, is the maximal set of objects that have all items $i \ (i \in I)$, which is expressed as $\text{¨}(I) = \{o \in U \mid \forall i \in I, (o, i) \in R\}$.

Definition 2.6 Intersection operation of two formal granules is denoted by $\otimes$, which is described as follows:

There are two formal granules $G_{\sim} = < I_{\sim}, \text{¨}(I_{\sim}) >$ and $G \ \ = < I \ \ , \text{¨}(I \ \ ) >$, respectively; then we have:

$G = < I, \text{¨}(I) > = G_{\sim} \otimes G \ \ = < I_{\sim} \cup I \ \ , \text{¨}(I_{\sim}) \cap \text{¨}(I \ \ ) >$.

3 A novel mining model

Firstly, we present some *definitions*, *properties*, *theorems*, *and corollaries* from the Galois connection and granular computing. And propose a novel model for mining frequent closed itemsets based on granule computing.

3.1 Basic concepts

Definition 3.1 Itemset support, for a formal context $D = (U, A, R)$, the support of the itemset I is expressed as $support(I) = □ \text{¨}(I) \ □ \ U□$.

Definition 3.2 Frequent itemsets, the itemset I is said to be frequent if the support of I in D is at least the given $minsupport$. The set FI of frequent itemsets in D is defined as $FI = \{I \subseteq A \mid support(I) \geq minsupport\}$.

Property 3.1 All subsets of a frequent itemset are frequent; all supersets of an infrequent itemset are infrequent. (Intuitive in [2])

Definition 3.3 Frequent closed itemsets, the closed itemset C is said to be frequent if the support of C in D is at least the given $minsupport$. The set FCI of frequent closed itemsets in D is defined as follows:

$FCI = \{C \subseteq A \mid C = h(C) \wedge support(C) \geq minsupport\}$.

Property 3.2 Frequent closed itemsets FCI is the subset of frequent itemset FI, namely $FCI \subseteq FI$.

Definition 3.4 The smallest frequent closed itemsets, the frequent itemset I is said to be the smallest frequent closed itemset if $\forall I^\circ \subset I, support(I) < support(I^\circ)$. The set FC_{min} of the smallest frequent closed itemsets in D is $FC_{min} = \{I \in FI \mid \forall I^\circ \subset I \wedge support(I) < support(I^\circ)\}$.

Theorem 3.1 For a formal context $D = (U, A, R)$, if I be a frequent closed itemset, and there is the smallest frequent closed itemset $I'(\text{¨}(I) = \text{¨}(I'))$, i.e.

$\forall \ I \in FCI \Rightarrow \exists I' \in FC_{min} \wedge \text{¨}(I) = \text{¨}(I')$.

Proof. Let $□ I \ \ \boxminus k$, there are two cases as follows:

(1) If $\forall \ I_1 \subset I \ (□ I_1 \ \ \boxminus k - 1)$, and have $support(I) < support(I_1) \Rightarrow \forall \ I^\circ \subset I_1 \subset I \wedge support(I) < support(I^\circ)$. Since $I \in FCI \subseteq FI$, we have $I \in FC_{min}$. Let $I' = I$, and we have $I' \in FC_{min} \wedge \text{¨}(I) = \text{¨}(I')$.

(2) If $\exists \ I_1 \subset I \ (□ I_1 \ \ \boxminus k - 1)$, and have $support(I) = support(I_1) \Rightarrow \text{¨}(I) = \text{¨}(I_1)$.

(i) If $\forall \ I_2 \subset I_1 \subset I \ (□ I_2 \ \ \boxminus k - 2)$, and $support(I_1) < support(I_2) \Rightarrow \forall I^\circ \subset I_2 \subset I_1 \wedge support(I_1) < support(I^\circ)$. Since $I_1 \subset I \in FCI \subseteq FI$, we have $I_1 \in FC_{min}$. Let $I' = I_1$, and we have $I' \in FC_{min} \wedge \text{¨}(I) = \text{¨}(I_1) = \text{¨}(I')$.

(ii) Otherwise $\exists \ I_2 \subset I_1 \subset I \ (□ I_2 \ \ \boxminus k - 2)$, and have $support(I_1) = support(I_2) \Rightarrow \text{¨}(I_1) = \text{¨}(I_2) ...$

Go on doing until the k^{th} step, and $\exists \ support(I) = support(I_k) \wedge I_k \in FC_{min}$. Let $I' = I_k$, and we have $I' \in FC_{min} \wedge \text{¨}(I) = \text{¨}(I_1) = ... = \text{¨}(I_k) = \text{¨}(I')$.

Based on definition 2.4 and theorem 3.1, we have:

Corollary 3.1 Let I be the smallest frequent closed itemset, i.e. $I \in FC_{min}$. And the frequent closed itemset corresponding to I is $h(I) = \mu(\ddot{}\,(I))$.

Corollary 3.2 For a formal context $D = (U, A, R)$, the set FCI of frequent closed itemsets in D is expressed as $FCI = \{h(I) \mid I \in FC_{min}\}$.

Theorem 3.2 Let $I_{\sim} \subseteq I \quad \subseteq A$, where $support(I_{\sim}) = support(I \quad)$. Then we have $h(I_{\sim}) = h(I \quad)$ and $\forall I \subseteq A$, $h(I_{\sim} \cup I) = h(I \quad \cup I)$.

Proof. $\because I_{\sim} \subset I \quad \subseteq A \wedge support(I_{\sim}) = support(I \quad)$

$\therefore \quad \Box\ddot{}\,(I_{\sim}) \boxminus \ddot{}\Box(I \quad)$

$\therefore \quad \ddot{}\,(I_{\sim}) = \ddot{}\,(I \quad)$

$\therefore \mu(\ddot{}\,(I_{\sim})) = \mu(\ddot{}\,(I \quad)),\ i.e.\ h(I_{\sim}) = h(I \quad)$

$\because I \subseteq A$

$\therefore h(I_{\sim} \cup I) = h(h(I_{\sim}) \cup h(I))$ (Theorem 2.1)

$\because h(I_{\sim}) = h(I \quad)$

$\therefore h(I_{\sim} \cup I) = h(h(I \quad) \cup h(I)) = h(I \quad \cup I)$.

Theorem 3.3 $I \in FC_{min} \Rightarrow \forall\ I^{\circ} \subset I \wedge I^{\circ} \in FC_{min}$.

Proof. Suppose $I \in FC_{min} \Rightarrow \exists\ I_1 \subset I \wedge I_1 \notin FC_{min}$.

$\because\ I_1 \subset I \wedge I_1 \notin FC_{min}$

$\therefore \exists\ I_2 \subset I_1 \wedge support(I_1) = support(I_2)$

$\therefore \quad \ddot{}\,(I_2) = \ddot{}\,(I_1)$

$\exists\ (I' = I - I_1) \wedge (I' \cup I_2 \subset I) \quad (I_3 = I' \cup I_2)$

$\therefore \quad \ddot{}\,(I') \supseteq \ddot{}\,(I_3) \supseteq \ddot{}\,(I)$

$\because I_3 \supset I_2,\ i.e.\ \ddot{}\,(I_3) \subseteq \ddot{}\,(I_2)$

$\therefore \quad \ddot{}\,(I_3) \subseteq \ddot{}\,(I_1)$

$\therefore \ddot{}\,(I_1) \cap \ddot{}\,(I') \supseteq \ddot{}\,(I_3)$

$\because \ddot{}\,(I) = \ddot{}\,(I_1 \cup I') = \ddot{}\,(I_1) \cap \ddot{}\,(I')$ (Definition 2.6)

$\therefore \ddot{}\,(I_3) \subseteq \ddot{}\,(I)$

$\therefore \ddot{}\,(I_3) = \ddot{}\,(I),\ i.e.\ support(I_3) = support(I)$

$\therefore \exists\ I_3 \subset I \wedge support(I_3) = support(I)$

$\therefore I \notin FC_{min}$. However, the itemset I is the smallest frequent closed itemset, namely $I \in FC_{min}$.

$\therefore I \in FC_{min} \not\Rightarrow \exists\ I_1 \subset I \wedge I_1 \notin FC_{min}$

$\therefore I \in FC_{min} \Rightarrow \forall\ I^{\circ} \subset I \wedge I^{\circ} \in FC_{min}$.

Corollary 3.3 $I \in FC_{min} \Rightarrow \forall\ I^{\circ} \subset I \wedge I^{\circ} \in FC_{min}$, $(\Box I^{\circ} \boxminus I\Box - 1)$

Definition 3.5 The smallest frequent closed granules set, the formal granule $G =< I, \ddot{}\,(I) >$ is said to be the smallest frequent closed granule G_{min} if the intension I of G is the smallest frequent closed itemset. The set FG_{min} of the smallest frequent closed granules is defined as:

$$FG_{min} = \{G =< I, \ddot{}\,(I) > \mid I \in FC_{min}\}$$

3.2 Frequent closed itemsets mining

In this section, we propose a novel model for mining frequent closed itemsets based on granule computing. Based on the previous introductions, the following is a formal statement of this model.

For a formal context $D = (U, A, R)$, discovering all frequent closed itemsets in D can be divided into two steps as follows:

(1)According to the minimal support given by user, mining the smallest frequent closed granules set in D . (Details in the steps from (1) to (18) from Section 4.2)

(2)Based on the smallest frequent closed granules set, discovering all frequent closed itemsets in D . (Details in the steps from (19) to (21) from Section 4.2)

Here the first step is based on definition 3.5, theorem 2.1, and theorem 3.2; the second step refers to Definition 2.4, Proposition 2.1, and Theorem 3.1(Corollary 3.1). From the theory, they provide the demonstration for the novel mining model.

4 The efficient mining algorithm

In this section, we use an efficient mining algorithm to describe the novel model, which is denoted by EMFCI.

4.1 Generator function

Here, we propose a function for generating the intension of the smallest frequent closed granules.

Definition 4.1 Set vector operation□ for two sets is defined as follows:

Let $P = \{p_1, p_2, ..., p_m\}, Q = \{q_1, q_2, ..., q_n\}$ be two sets, and then the set vector operation is expressed as $P^T \Box\ Q$

$$= \begin{pmatrix} \{p_1\} \\ \{p_2\} \\ ... \\ \{p_m\} \end{pmatrix} \Box \begin{pmatrix} \emptyset & \{q_1\} & \{q_2\} & ... & \{q_n\} \end{pmatrix}$$

$$= \begin{pmatrix} \{p_1\} & \{p_1, q_1\} & \{p_1, q_2\} & ... & \{p_1, q_n\} \\ \{p_2\} & \{p_2, q_1\} & \{p_2, q_2\} & ... & \{p_2, q_n\} \\ ... & ... & ... & ... & ... \\ \{p_m\} & \{p_m, q_1\} & \{p_m, q_2\} & ... & \{p_m, q_n\} \end{pmatrix}$$

$= \{\{p_1\}, \{p_1, q_1\}, \{p_1, q_2\}, ..., \{p_1, q_n\}, \{p_2\}, \{p_2, q_1\},$
$\{p_2, q_2\}, ..., \{p_2, q_n\}, ..., \{p_m\}, \{p_m, q_1\}, \{p_m, q_2\},$
$..., \{p_m, q_n\}\}$ (Formal notation)

$= \{\{p_1\}, \{p_1q_1\}, \{p_1q_2\}, ..., \{p_1q_n\}, \{p_2\}, \{p_2q_1\}, \{p_2q_2\},$
$..., \{p_2q_n\}, ..., \{p_m\}, \{p_mq_1\}, \{p_mq_2\}, ..., \{p_mq_n\}\}$.
(Simple notation)

The operation is the main idea of generator function, let P, Q be two sets, it is expressed as $f(P, Q) = P^T \Box\ Q$. The application of $f(P, Q)$ refers to Section 4.2.

For example, for a formal context $D = (U, A, R)$, let A be a general itemset $\{a, b, c\}$, and then we use the set vector operation to generate $P(A)$ $(\forall p \in P(A) \wedge p \neq \emptyset)$ as follows:

(1) $P(A) = \emptyset$;

(2) $I_x = \{a\} \Rightarrow P(A) = P(A) \cup (I_x^T \Box\ P(A))$
$= (\{a\}) \Box\ (\emptyset) = \{\{a\}\}$;

$$(3)\, I_x = \{b\} \Rightarrow P(A) = P(A) \cup (I_x^T \,□\, P(A))$$
$$= \{\{a\}\} \cup ((\{b\}) □ \,(\emptyset \quad \{a\}))$$
$$= \{\{a\},\{b\},\{ab\}\};$$
$$(4)\, I_x = \{c\} \Rightarrow P(A) = P(A) \cup (I_x^T \,□\, P(A))$$
$$= \{\{a\},\{b\},\{ab\}\} \cup ((\{c\}) □ \,(\emptyset \quad \{a\} \quad \{b\} \quad \{ab\}))$$
$$= \{\{a\},\{b\},\{ab\},\{c\},\{ac\},\{bc\},\{abc\}\}.$$

For a formal context $D = (U, A, R)$, if A is a general itemsets, namely, it is a set of Boolean attributes, $P(A)$ is general the power set where □ $P(A)$ ⊟ $2^{□A□} - 1$. But if A is a set of quantitative attributes, where $P(A)$ is called the extended power set of A, and □ $P(A)$ is expressed as:

□ $P(A)$ ⊟ $\prod_{a \in A} (V_a □ + 1) - 1$, here V_a is a reprocessed discrete range of attribute $a \in A$.

4.2 An algorithm for mining frequent closed itemsets

In this section, we describe the efficient algorithm based on the novel model in Section 3 via the following pseudo code.

Algorithm: EMFCI

Input: a formal context $D = (U, A, R)$, the minimal support $minsupport$.

Output: frequent closed itemsets FCI.

```
(1)Read $D$;
(2)Construct $FG = \{FG_a \mid a \in A \wedge \forall\ G =< I, ¨(I) >\in FG_a \wedge I \subset V_a \wedge □I ⊟ 1 \wedge ¨□(I) \geq minsupport\}$;
(3) $F = \{F_a \subset V_a \mid \forall v \in F_a \wedge G =< \{v\}, ¨(\{v\}) >\in FG_a \wedge FG_a \in FG \wedge a \in A\}$; //$V_a$ is the range of attribute $a \in A$.
(4) $FC_{min} = \emptyset$;
(5)For $(\forall \sim\ \in F)$ do begin
(6) $S_c = \sim □\ FC_{min}$; //Generate the candidate
(7) For $(\forall s \in S_c)$ do begin
(8)  If $((\forall t_1 \in N_{FI} \wedge t_1 \not\subset s) \wedge (\forall t_2 \in N_{FCmin} \wedge t_2 \not\subset s))$ then
(9)       Construct $G =< s, ¨(s) >$;
(10)       If $(□¨(s)$ ⊵ $minsupport)$ then
(11)           If $(\forall t \subset s \wedge □¨(s))$ ⊴ $¨□(t))$ ) then
(12)               Write $G =< s, ¨(s) >$ to $FG_{min}$;
(13)               Write $s$ to $FC_{min}$;
(14)           else
(15)               Write $s$ to $N_{FCmin}$;
(16)     else
(17)             Write $s$ to $N_{FI}$;
(18)End
(19)For $(\forall G =< I, ¨(I) >\in FG_{min})$ do begin
(20)   Write $h(I) = \mu(¨(I))$ to $FCI$;
(21)End
(22)Answer $FCI$;
```

These steps from (1) to (18) in the algorithm extract the smallest frequent closed granules set. And these steps from (19) to (21) generate all frequent closed itemsets.

4.3 Example and analysis

Here, we firstly provide an example for the algorithm, and then analyse the pruning strategies in the algorithm.

No.	Operation
1	$FG = \{<\{a\},\{1,3,5\}>, <\{b\},\{2,3,4\}>,$ $<\{c\},\{1,2,5\}>, <\{e\},\{3,4,5\}>\}$ (Pruning $\{d\}$ by property 3.1and definition 3.3)
2	$F = \{\{a\},\{b\},\{c\},\{e\}\}$
3	$\sim = \{a\} \Rightarrow S_c = \{\{a\}\}$ $FG_{min} = \{<\{a\},\{1,3,5\}>\}$, $FC_{min} = \{\{a\}\}$
4	$\sim = \{b\} \Rightarrow S_c = \{\{b\},\{ab\}\}$ $FG_{min} = \{<\{a\},\{1,3,5\}>, <\{b\},\{2,3,4\}>\}$ $FC_{min} = \{\{a\},\{b\}\}$ (Pruning $\{ab\}$ by property 3.1and definition 3.3)
5	$\sim = \{c\} \Rightarrow S_c = \{\{c\},\{ac\},\{bc\}\}$ $FG_{min} = \{<\{a\},\{1,3,5\}>, <\{b\},\{2,3,4\}>\}$ $<\{c\},\{1,2,5\}>, <\{ac\},\{1,5\}>\}$ $FC_{min} = \{\{a\},\{b\},\{c\},\{ac\}\}$ (Pruning $\{bc\}$ by property 3.1and definition 3.3)
6	$\sim = \{e\} \Rightarrow S_c = \{\{e\},\{ae\},\{be\},\{ce\},\{ace\}\}$ $FG_{min} = \{<\{a\},\{1,3,5\}>, <\{b\},\{2,3,4\}>\}$ $<\{c\},\{1,2,5\}>, <\{ac\},\{1,5\}>, <\{e\},\{3,4,5\}>,$ $<\{ae\},\{3,5\}>, <\{be\},\{3,4\}>\}$ $FC_{min} = \{\{a\},\{b\},\{c\},\{ac\},\{e\},\{ae\},\{be\}\}$ (Pruning $\{ce, ace\}$ by property 3.1and definition 3.3)` Note: the search course is ended, discovering all the smallest frequent closed granules FC_{min}
7	$h(\{a\}) = \{u_1 \cap u_3 \cap u_5\} = \{a\}$ $h(\{b\}) = \{u_2 \cap u_3 \cap u_4\} = \{b\}$ $h(\{c\}) = \{u_1 \cap u_2 \cap u_5\} = \{c\}$ $h(\{ac\}) = \{u_1 \cap u_5\} = \{ac\}$ $h(\{e\}) = \{u_3 \cap u_4 \cap u_5\} = \{e\}$ $h(\{ae\}) = \{u_3 \cap u_5\} = \{ae\}$ $h(\{be\}) = \{u_3 \cap u_4\} = \{be\}$ Note: based on the smallest frequent closed granules set FC_{min}, getting all frequent closed itemsets
8	Answer $FCI = \{\{a\},\{b\},\{c\},\{ac\},\{e\},\{ae\},\{be\}\}$

Table 1: Frequent closed itemsets mining for $minsupport = 40\%$.

For a formal context $D=(U,A,R)$, where $A=\{a,b,c,d,e\}, U=\{u_1,u_2,u_3,u_4,u_5\}, u_1=\{acd\}, u_2=\{bc\}, u_3=\{abe\}, u_4=\{be\}, u_5=\{ace\}$; and $minsupport = 40\%$. The course of discovering frequent closed itemsets is described as table 1.

For mining frequent closed itemsets, the algorithm adopts some pruning strategies as follows, property 3.1, definition 3.3 and 3.4, and theorem 3.3. They can help the algorithm efficiently reduce the search space for mining frequent closed itemsets.

5 Performance and scalability study

In this section, we design the following experiments on these different datasets:

Firstly, we report the performances of the algorithm EMFCI with A-Close and CLOSET on the six different datasets.

Secondly, we report the relationships between some parameters of the datasets and the performances of the algorithm EMFCI for mining frequent closed itemsets.

Finally, for the bottleneck of the algorithm EMFCI, we improve it to get the algorithm IEMFCI, and report its performances on the extended high dimension dataset to show the scalability of the algorithm EMFCI.

There are two original datasets as follows:

The first is the Food Mart 2000 retail dataset, which comes from SQL Server 2000. It contains 164558 records in 1998. By the same customer at the same time as a basket, we take items purchased from these records. Because the supports of the bottom items are small, we generalize the bottom items to **the product department**. Finally, we obtain 34015 transactions with time-stamps. It is a dataset with the **Boolean** attributes.

The second is from a Web log data, which is a real data that expresses some behaviour of students browsing, where the attributes set is made of *login time, duration, network flow, IDtype, and sex*. The dataset with the **discrete quantitative** attributes has 296031 transactions.

Now, we generalize attributes, and replicate some attributes or transactions to create the following extended datasets described as table 2, where each dataset can be defined as a formal mining context $D=(U,A,R)$.

All the experiments are performed on an Intel (R) Core (TM)2 Duo CPU (T6570 @) 2.10 GHz 1.19GHz) PC with 1.99 GB main memory, running on Microsoft Window XP Professional. All the programs are written in C# with Microsoft Visual Studio 2008. The algorithm A-close and CLOSET are implemented as described in [3] and [6].

Name	Descriptions	$|P(A)|$; $|U|$
Dataset 1	The first original dataset	$2^{22}-1$; 34015
Dataset 2	Replicating dataset 1 three attributes	$2^{25}-1$; 34015
Dataset 3	Replicating dataset 1 four times	$2^{22}-1$; 5*34015
Dataset 4	The second original dataset	5*4*4*14*3-1; 296031
Dataset 5	Replicating dataset 1 one attribute	5*4*4*14*3*5-1; 296031
Dataset 6	Replicating dataset 4 one time	5*4*4*14*3-1; 2*296031
Dataset 7	For the Food Mart 2000, we regard the same customer at the same time as a basket and generalize the bottom items to **the product subcategory**	$2^{102}-1$; 34015

Table 2: The datasets used in the experiments.

5.1 The experiments of performance comparison

In this section, for discovering frequent closed itemsets on these different datasets, we compare the algorithm EMFCI with the algorithm A-close and CLOSET from the following two aspects, namely, one is comparing the performances among them as the minimal support is added; the other is comparing them as the number of frequent closed itemsets is added.

1. Testing on the original datasets

For the two original datasets, we firstly compare the algorithm EMFCI with the A-close and CLOSET based on the varying minimal support and the number of frequent closed itemsets. These experimental results are described as figure 1, 2, 3, and 4, respectively.

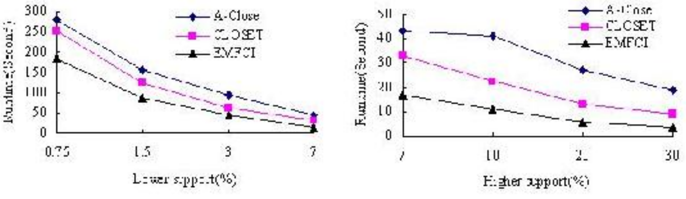

Figure 1: Performance comparison with the support on dataset 1.

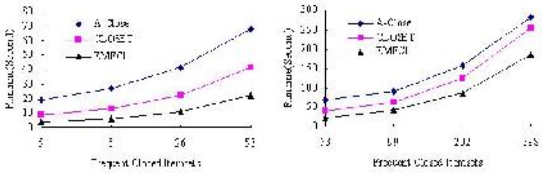

Figure 2: Performance comparison with the number of frequent closed itemsets on dataset 1.

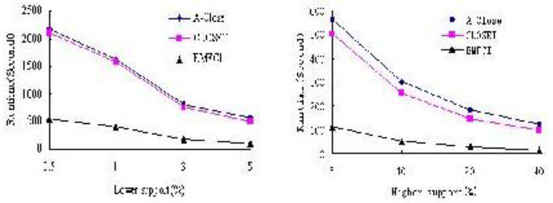

Figure 3: Performance comparison with the support on dataset 4.

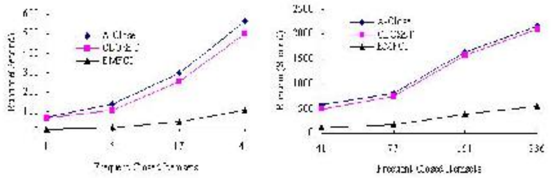

Figure 4: Performance comparison with the number of frequent closed itemsets on dataset 4.

Based on the comparison results from figure 1, 2, 3, and 4, we know that the performances of the algorithm EMFCI are better than the A-close and CLOSET.

Obviously, the algorithm CLOSET is also superior to the A-close. Hence, we don’t compare the EMFCI with the A-close in the following experiments.

2. Testing on the extended datasets

We further report the performances of the algorithm EMFCI on the extended datasets. Based on the different minimal support and the number of frequent closed itemsets, we compare the EMFCI with the CLOSET, the experimental results are described as figure 5 to 12.

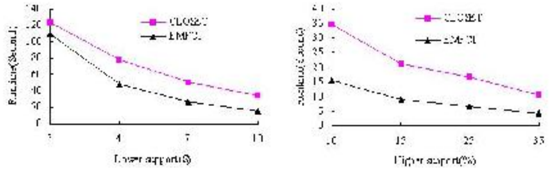

Figure 5: Performance comparison with the support on dataset 2.

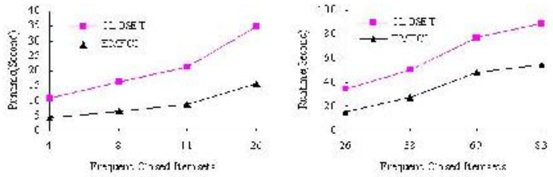

Figure 6: Performance comparison with the number of frequent closed itemsets on dataset 2.

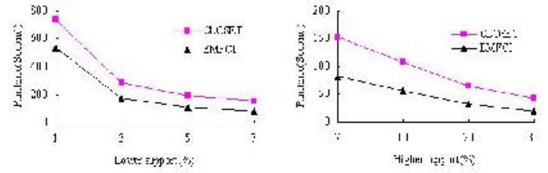

Figure 7: Performance comparison with the support on dataset 3.

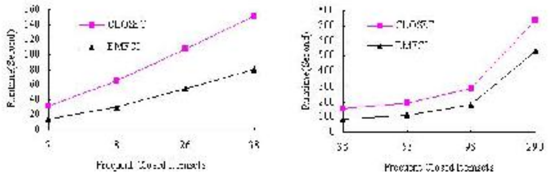

Figure 8: Performance comparison with the number of frequent closed itemsets on dataset 3.

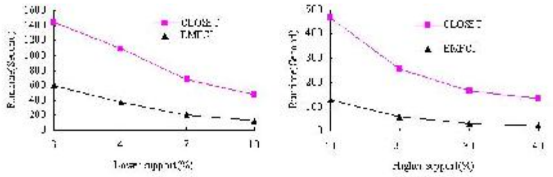

Figure 9: Performance comparison with the support on dataset 5.

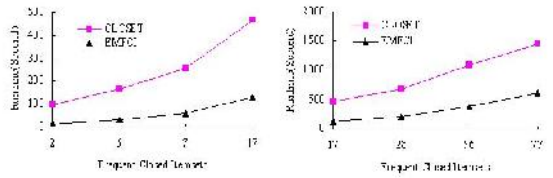

Figure 10: Performance comparison with the number of frequent closed itemsets on dataset 5.

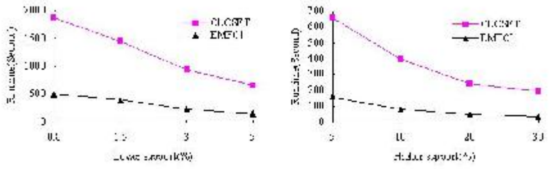

Figure 11: Performance comparison with the support on dataset 6.

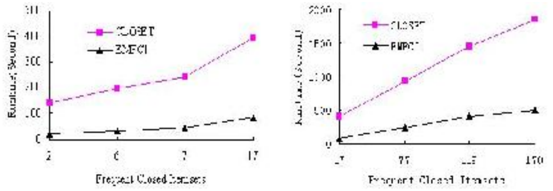

Figure 12: Performance comparison with the number of frequent closed itemsets on dataset 6.

Based on the comparison results from figure 5 to 12, we know that the performances of the algorithm EMFCI are also better than the CLOSET on the datasets with the Boolean or quantitative attributes.

5.2 The relationships between these parameters and performances

In this part, we mainly discuss the relationships between the performances and the following parameters:

$|U|$, is the number of objects in the formal mining context $D = (U, A, R)$, in other word, it is the number of transactions in the mining database.

□$P(I)$ [, is the number of nonempty power sets for attribute values, called the **search space** of the algorithm, where I is the smallest frequent closed itemsets from the attribute set A, $P(I)$ is defined as the power set of I. (Refer to section 4.1)

Here, the representation of the performances has two kinds of parameters as follows:

$t(x)$: is the runtime of algorithm x, which is from input to output for mining frequent closed itemsets.

p, is defined as the improved ratio of the runtime between the algorithm EMFCI and CLOSET, which is denoted by the following equation:

$p = 1 - t(EMFCI) / t(CLOSET)$.

1. The relationships between the performances and the search space

(1)Reporting the relationships on the extended dataset of the first original dataset

For the first original dataset, namely, dataset 1, we test the trend of the performances as the search space is increasing on dataset 2, which is the extended dataset with replicating three attributes of the first dataset. As the search space is varying, the trend of the runtime for the algorithm EMFCI is expressed as figure 13, the trend of the improved ratio between the algorithm EMFCI and CLOSET is expressed as figure 14.

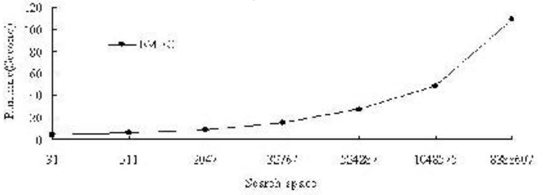

Figure 13: The trend of the runtime on dataset 2.

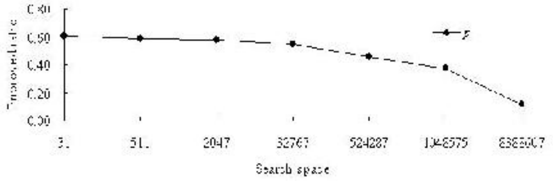

Figure 14: The trend of the improved ratio on dataset 2.

Based on figure 13, we know that the runtime is added as the search space is increasing. Based on figure 14, we find that the improved ratio is reduced as the search space is increasing.

(2)Reporting the relationships on the extended dataset of the second original dataset

For the second original dataset, namely, dataset 4, we extend an attribute to get dataset 5, and test the trend of the performances on the dataset. The experimental results are expressed as figure 15 and 16, respectively.

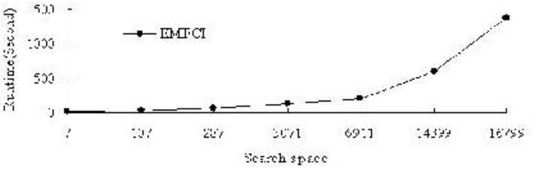

Figure 15: The trend of the runtime on dataset 5.

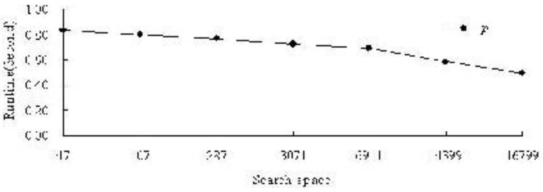

Figure 16: The trend of the improved ratio on dataset 5.

According to figure 15 and 16, we get the similar comparisons results as above. Hence, we can draw the following conclusions:

The runtime of the algorithm EMFCI is added as the search space is increasing; on the contrary, the improved ratio is reduced. Namely, if the search space is increasing, the performances of the algorithm EMFCI will become worse and worse. In other word, the algorithm is not suitable for mining the dataset with too many smallest frequent closed itemsets.

2. The relationships among the performances, the search space and the number of objects

(1)Reporting the relationships on the first original dataset and its extended dataset

For the first original dataset (dataset 1), and its extended dataset, dataset 3 with replicating its objects four times, we test the trend of the performances as the search space is increasing on the two datasets. As the search space is varying, the trend of the runtime for the algorithm EMFCI is expressed as figure 17, the trend of the improved ratio between the algorithm EMFCI and CLOSET is expressed as figure 18.

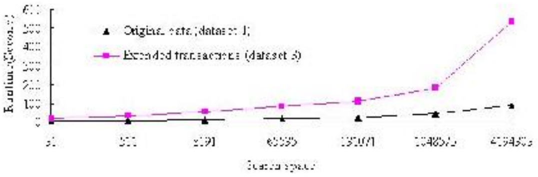

Figure 17: The trend of the runtime on dataset 1 and 3.

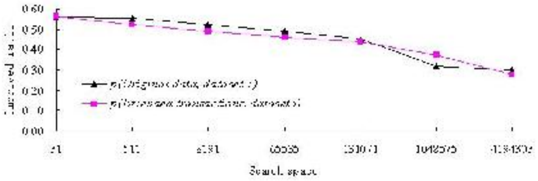

Figure 18: The trend of the improved ratio on dataset 1 and 3.

Based on figure17, we know that the runtime of the algorithm is added as the search space or the number of objects is increasing.

Based on figure18, we find that the improved ratio of the algorithm is reduced as the search space is increasing, but it become relatively stable as the number of objects is increasing.

(2)Reporting the relationships on the second original dataset and its extended dataset

For the second original dataset, namely, dataset 4, we replicate its objects one time to get dataset 6, and test the trend of the performances on the dataset 4 and 6. The experimental results are expressed as figure 19 and 20, respectively.

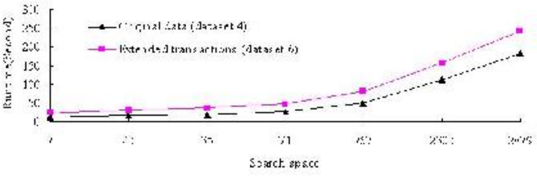

Figure 19: The trend of the runtime on dataset 4 and 6

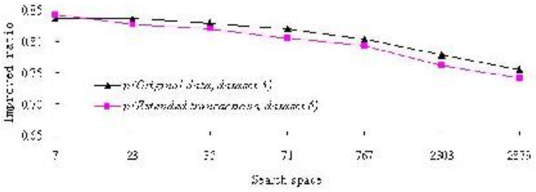

Figure 20: The trend of the improved ratio on dataset 4 and 6

According to figure 19 and 20, we draw the same conclusions as follows:

The runtime of the algorithm EMFCI is added as the search space or the number of objects is increasing, the improved ratio of the algorithm is reduced as the search space is increasing, but it become relatively stable as the number of objects is adding. Namely, the performances of the algorithm EMFCI will become relatively stable as the number of objects is increasing. Hence, it is suitable for mining dynamic transactions datasets.

According to all these experimental results, we can **draw the following conclusions**:

(1) The performances of the algorithm EMFCI are better than the traditional typical algorithms for mining frequent closed itemsets on the datasets with the Boolean attributes or the 1uantitative attributes.

(2) The runtime of the algorithm EMFCI is added as the search space. If the search space is too large, its performances will become worse and worse. This is the bottleneck of the algorithm.

(3) The runtime of the EMFCI is also added as the number of objects is increasing.

(4) For the algorithm CLOSET, the improved ratio of the algorithm is reduced as the search space is adding, but it become relatively stable as the number of objects is increasing. Namely, the performances of the EMFCI will become relatively stable as the number of objects is increasing. It is suitable for mining dynamic transactions datasets.

5.3 A further discussion for solving the bottleneck of the algorithm

Based on these conclusions in section 5.2, for the formal mining context $D = (U, A, R)$, if the search space□$P(I)$ [is overlarge, where $I(I \subseteq A)$ is the smallest frequent closed itemsets, $P(I)$ is defined as the power set of I, the performance of EMFCI will become worse and worse.

In this section, we adopt **a partitioning method** to avoid the bottleneck. In other word, the overlarge search space is divided into some smaller search spaces. The theoretical basis can be described as follows:

Let $I = \{a^{t_1}, a^{t_2}, ..., a^{t_m}\}(I \subseteq A)$, and then we have the following $\| P(I) \| = \prod_{i=1}^{m} (\| V_{a^{t_1}} \| + 1) - 1$, namely,

$$\| P(I) \| + 1 = \prod_{i=1}^{m} (\| V_{a^{t_1}} \| + 1) =$$

$$\underbrace{(\| V_{a^{t_1}} \| + 1) \cdot (\| V_{a^{t_2}} \| + 1) \cdot ... \cdot (\| V_{a^{t_{m_1}}} \| + 1)}_{m_1} \cdot$$

$$\underbrace{(\| V_{a^{t_{m_1+1}}} \| + 1) \cdot (\| V_{a^{t_{m_1+2}}} \| + 1) \cdot ... \cdot (\| V_{a^{t_{m_1+m_2}}} \| + 1)}_{m_2} \cdot ... \cdot$$

$$\underbrace{(\| V_{a^{t_{m_1+m_2+...+m_{(k-1)}+1}}} \| + 1) \cdot ... \cdot (\| V_{a^{t_{m_1+m_2+...+m_{(k-1)}+m_k}}} \| + 1)}_{m_k};$$

$(m_1 + m_2 + ... + m_k = m)$.

Obviously, we also have $\| P(I) \| + 1 =$

$(\| P(I_{m_1}) \| + 1) \cdot (\| P(I_{m_2}) \| + 1) \cdot ... \cdot (\| P(I_{m_k}) \| + 1)$;

Where $I_{m_1} = \{a^{t_1}, a^{t_2}, ..., a^{t_{m_1}}\}$,

$I_{m_2} = \{a^{t_{m_1+1}}, a^{t_{m_1+2}}, ..., a^{t_{m_1+m_2}}\}, ...,$

$I_{m_k} = \{a^{t_{m_1+m_2+...+m_{(k-1)}+1}}, ..., a^{t_{m_1+m_2+...+m_{(k-1)}+m_k}}\}$.

In this paper, we let $\| P(I_{m_i}) \| <$ ª $= 2^{19}$. If ª is too big, the method also has the same bottleneck; if ª is too small, the cost of partitioning search space is expensive. For these two cases, their performances are expressed as figure 23.

The partitioning method is used in the algorithm EMFCI, which is called improved EMFCI, i.e. IEMFCI.

5.3.1 Example

For the example in section 4.3, we use the algorithm IEMFCI to discover frequent closed itemsets, the course of which is described as follows, where ª $= 4$.

(**Note:** ª $= 4$ used in the example, ª $= 2^{19}$ used in the following experiments)

Step1. $FG = \{<\{a\}, \{1,3,5\}>, <\{b\}, \{2,3,4\}>,$
$<\{c\}, \{1,2,5\}>, <\{e\}, \{3,4,5\}>\}$.

Step2. $F = \{\{a\}, \{b\}, \{c\}, \{e\}\}, \| P(F) \| = 15 >$ ª $= 4$.

Step3. **Partitioning** the search space, get two search spaces $F_1 = \{\{a\}, \{b\}\}, F_2 = \{\{c\}, \{e\}\}$, where $\| P(F_i) \| < 4$.

Step4. For the first search space $F_1 = \{\{a\}, \{b\}\}$, have

① $\sim = \{a\} \Rightarrow S_c = \{\{a\}\}$

$FG_{min}^1 = \{<\{a\}, \{1,3,5\}>\}$, $FC_{min}^1 = \{\{a\}\}$;

② $\sim = \{b\} \Rightarrow S_c = \{\{b\}, \{ab\}\}$

$FG_{min}^1 = \{<\{a\}, \{1,3,5\}>, <\{b\}, \{2,3,4\}>\}$,

$FC_{min}^1 = \{\{a\}, \{b\}\}$.

For the second search space $F_2 = \{\{c\}, \{e\}\}$, have

① $\sim = \{c\} \Rightarrow S_c = \{\{c\}\}$

$FG_{min}^2 = \{<\{c\}, \{1,2,5\}>\}$, $FC_{min}^2 = \{\{c\}\}$;

② $\sim = \{e\} \Rightarrow S_c = \{\{e\},\{ce\}\}$

$FG^2_{min} = \{<\{c\},\{1,2,5\}>,<\{e\},\{3,4,5\}>\}$,

$FC^2_{min} = \{\{c\},\{e\}\}$.

Step5. $F = \{FC^1_{min}, FC^2_{min}\}$, repeating the step2, where $\| P(F) \| = 15 > 4$, but $\| F \| = 2$, the partitioning operation must be ended; otherwise, the algorithm need to continue to partition the search space.

$\sim = FC^1_{min} \Rightarrow S_c = \{\{a\},\{b\}\}$,

$FG_{min} = \{<\{a\},\{1,3,5\}>,<\{b\},\{2,3,4\}>\}$,

$FC_{min} = \{\{a\},\{b\}\}$;

$$\sim = FC^2_{min} \Rightarrow S_c = \begin{pmatrix}\{c\}\\ \{e\}\end{pmatrix}\left(\emptyset \quad \{a\} \quad \{b\}\right) =$$

$$\{\{c\},\{ac\},\{bc\},\{e\},\{ae\},\{be\}\};$$

$FG_{min} = \{<\{a\},\{1,3,5\}>,<\{b\},\{2,3,4\}>\}$

$<\{c\},\{1,2,5\}>,<\{ac\},\{1,5\}>,<\{e\},\{3,4,5\}>,$

$<\{ae\},\{3,5\}>,<\{be\},\{3,4\}>\}$

$FC_{min} = \{\{a\},\{b\},\{c\},\{ac\},\{e\},\{ae\},\{be\}\}$

The rest of steps are the same as the example in section 4.3. The algorithm IEMFCI reduces the checking of itemset $\{ace\}$, but adds the task of partitioning. As the number of transactions is lesser, the example does not show its advantage, please see the experiments in section 5.3.3. Here, the example only describes the execution course of IEMFCI.

5.3.2 Comparisons of the time and space complexity

For $D = (U, A, R)$, let C be a set of frequent closed itemsets, and let L be the average length of frequent closed itemsets, $k \geq 2$ is a parameter with partitioning the search space. The comparisons are expressed as table 3.

Items	Time complexity	Space complexity
A-close	$O(\|C\|^L)$	$O(\|C\| / \|A\|)$
CLOSET	$O(\|C\|^2)$	$O(\|C\|)$
IEMFCI	$O((L/k+1) \cdot \|C\|)$	$O(\|C\| / k \cdot \|A\|)$

Table 3: Comparisons of the time and space complexity.

5.3.3 Test on the high dimension datasets

In this section, to show the scalability of the algorithm EMFCI, firstly, we compare the improved algorithm IEMFCI with EMFCI, A-close and CLOSET on the high dimension dataset (dataset 7 as table 1), which is an extended dataset based on the first original dataset. The comparison results are expressed as figure 21 and 22, where the parameter $p(2,m) = 2^m$ on the abscissa shows the search space □$P(I)$ □of the given support.

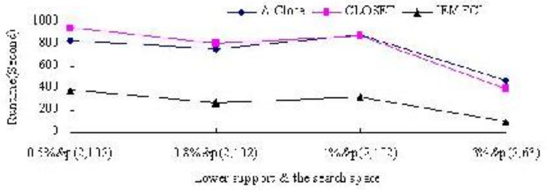

Figure 21: Performance comparison with the lower support on dataset 7.

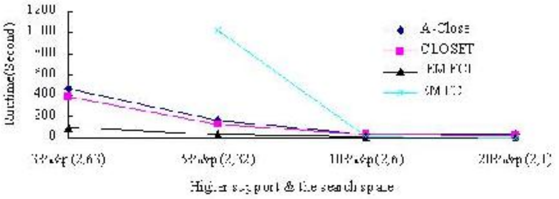

Figure 22: Performance comparison with the higher support on dataset 7.

Then, for the improved algorithm IEMFCI, we adopt different parameters ª to test its trend of performance, where ª $= 2^5$, ª $= 2^{19}$ and ª $= 2^{22}$. The comparison result is expressed as figure 23, where IEMFCI (ª $= p(2,n)$) is the improved algorithm IEMFCI when the parameter of partitioning the search space is ª $= p(2,n) = 2^n$.

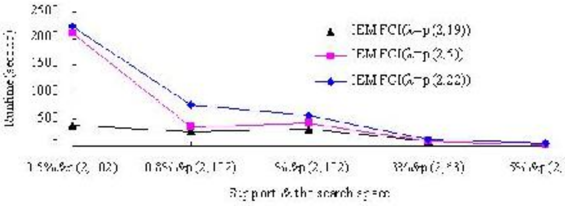

Figure 23: The trend of performance with the different parameter on dataset 7.

Based on these comparisons, we draw the following conclusions:

Firstly, the improved algorithm IEMFCI is better than the algorithms EMFCI, A-close and CLOSET.

Secondly, the improved algorithm IEMFCI gets rid of the bottleneck in the algorithms EMFCI, especially, when the search space□$P(I)$ □is overlarge, the advantage of IEMFCI is very distinct.

Finally, for the improved algorithm IEMFCI, the parameter of partitioning the search space is not too big, but it is not too small.

6 Conclusion

In this paper, for the shortcomings of typical algorithms for mining frequent closed itemsets, we propose an efficient algorithm for mining frequent closed itemsets, which is based on Galois connection and granular computing. We present the notion of smallest frequent closed granule to reduce the costed I/O for discovering frequent closed itemsets. And we propose a connection function for generating the smallest frequent closed itemsets in the enlarged frequent 1-item manner to reduce the costed CPU and the occupied main memory. But the number of the smallest frequent closed itemsets is too many, the performances of the algorithm become worse and worse, so we further discuss how to solve the bottleneck, namely, propose its improved algorithm on high dimension dataset. The algorithm is also suitable for mining dynamic transaction datasets.

Acknowledgement

The authors would like to thank the anonymous reviewers for the constructive comment. This work was a project supported by Chongqing Cutting-edge and Applied Foundation Research Program (Grant No. cstc2014jcyjA40035). And it was also supported by Scientific and Technological Research Program of Chongqing Three Gorges University (Grant No.13ZD20).

References

[1] R. Agrawal, T. Imielinski, and A. Swami (1993). Mining association rules between sets of items in large databases. *In Proceedings of the 1993 ACM SIGMOD Int'l Conference on Management of Data*, Washington DC, USA, pp. 207–216.

[2] R. Agrawal and R. Srikant (1994). Fast algorithms for mining association rules. *In Proceedings of the 20th Int'l Conference on Very large Data Bases*, Santiago, Chile, pp. 487–499.

[3] N. Pasquier, Y. Bastide and R. Taouil et al. (1999). Discovering frequent closed itemsets for association rules. *In Proceedings of the 7th Int'l Conference on Database Theory*, Jerusalem, Israel, January, pp. 398–416.

[4] Mohammed J. Zaki, Ching-Jui Hsiao (1999). Charm: An efficient algorithm for closed association rule mining. *Technical Report 99-10, Computer Science*, Rensselaer Polytechnic Institute.

[5] J. Han, J. Pei, and Y. Yin (2000). Mining frequent patterns without candidate generation. *In Proceedings of the 2000 ACM SIGMOD Int'l Conference on Management of Data*, New York, USA, pp. 1–12.

[6] J. Pei, J. Han, and R. Mao (2000). CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets. *In Proceedings of the 2000 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*. Dallas, Texas, USA, pp. 21–30.

[7] D. Burdick, M. Calimlim, and J. Gehrke (2001). MAFIA: A maximal frequent item set algorithm for transactional databases. *In Proceedings of the 17th Int'l Conference on Data Engineering*. Heidelberg, pp. 443-452.

[8] J. Y. Wang, J. Han, and J. Pei (2003). CLOSET+: Searching for the best strategies for mining frequent closed itemsets. *In Proceedings of the 9th ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining,* Washington, DC, pp. 236 - 245.

[9] C. Lucchese, S. Orlando, and R. Perego (2006). Fast and memory efficient mining of frequent closed itemsets. *IEEE Trans on Knowledge and Dada Engineering*, vol. 18, no. 1, pp. 21- 36.

[10] R. Singh, T. Johnsten, and V. Raghavan et al (2010). Efficient Algorithm for Discovering Potential Interesting Patterns with Closed Itemsets. *In Proceedings of the 2010 IEEE Int'l Conference on Granular Computing*, pp. 414 - 419.

[11] F. Nori, M. Deypir, and M. Hadi et al. (2011). A new sliding window based algorithm for frequent closed itemset mining over data streams. *In Proceedings of the 1st Int'l Conference on Computer and Knowledge Engineering*, IEEE Press, pp. 249-253.

[12] Guang-Peng Chen, Yu-Bin Yang, and Yao Zhang (2012). MapReduce-Based Balanced Mining for Closed Frequent Itemset. *In Proceedings of the 2012 IEEE 19th Int'l Conference on Web Services*, IEEE Press, pp. 652 - 653.

[13] M. Sreedevi, Reddy L.S.S. (2013). Mining regular closed patterns in transactional databases. *In Proceedings of the 2013 7th Int'l Conference on Intelligent Systems and Control*, IEEE Press, pp. 380 - 383.

[14] Yu-quan Z. and Yu-qing S. (2007). Research on an Algorithm for Mining Frequent Closed Itemsets. *Journal of Computer Research and Development*, vol. 44, no. 7, pp. 1177-1183.

[15] Shengwei L., Lingsheng L., and Chong H. (2009). Mining closed frequent itemset based on FP-Tree. *In Proceedings of the IEEE Int'l Conference on Granular Computing*, IEEE Press, pp. 354 - 357.

[16] Wachiramethin J., Werapun J. (2009). BPA: A Bitmap-Prefix-tree Array data structure for frequent closed pattern mining. *In Proceedings of the 2009 Int'l Conference on Machine Learning and Cybernetics*, IEEE Press, vol .1, pp. 154 - 160.

[17] Z. Pawlak (1982). Rough sets. *Journal of computing and information science in Engineering*, no.11, pp. 341–356.

[18] R. Wille (1982). Restructuring lattice theory: an approach based on hierarchies of concepts. *In: I. Rival (Ed.), Ordered Sets*, Reidel, Dordrecht-Boston, pp. 445–470.

[19] B. Ganter, R. Wille (1999). Formal Concept Analysis, Mathematic Foundations. *Springer*, Berlin.

[20] J. Poelmans, D. I. Ignatov, and S. O. Kuznetsov et al (2013). Formal concept analysis in knowledge processing: A survey on applications. *Expert Systems with Applications*, vol.40, no. 16, pp. 6538–6560.

[21] M. W. Shao, Y. Leung (2014). Relations between granular reduct and dominance reduct in formal contexts. *Knowledge-Based Systems*, vol.65, pp. 1–11.

[22] Wei-Zhi W., Yee Leung, Ju-Sheng M. (2009). Granular Computing and Knowledge Reduction in Formal Contexts. *IEEE Transactions on Knowledge and Data Engineering*, vol.21, no.10, pp. 1461-1474.

[23] R. Belohlavek, B. D. Baets, J. Konecny (2014). Granularity of attributes in formal concept analysis. *Information Sciences*, vol.260, pp.149–170.

[24] Hobbs J. R. (1985). Granularity. *In Proceedings of the 9th International Joint Conference on Artificial Intelligence*, San Francisco, USA, pp. 432-435.

[25] Giunchglia F., Walsh T. (1992). A theory of abstraction. *Artificial Intelligence*, vol. 57, no. 2-3. pp. 323-389.

[26] Yao Y.Y. (2004). A partition model of granular computing. *Lecture Notes in Computer Science Transactions on Rough Sets*, vol. 3100, pp.232–253.

[27] T. R. Qiu, X. Q. Chen, and Q. Liu et al. (2010). Granular Computing Approach to Finding Association Rules in Relational Database. *International Journal of intelligent systems*, no. 25, pp. 165–179.

[28] G. Fang, Y. Wu (2013). Frequent Spatiotemporal Association Patterns Mining Based on Granular Computing. *Informatica*, vol.37, no.4, pp.443-453.

[29] Pawlak Z. (1998). Granularity of knowledge, indiscernibility and rough sets. *In Proceedings of IEEE Int Conf on Fuzzy Systems*, IEEE Press, Anchorage, AK, pp.106–110.

[30] Zhang L., Zhang B. (2003). The quotient space theory of problem solving. *Lecture Notes in Computer Science*, vol. 2639, pp. 11–15.

An Approach for Context-based Reasoning in Ambient Intelligence

Hristijan Gjoreski
Department of Intelligent Systems, Jožef Stefan Institute
Jamova 39, Ljubljana, Slovenia

Thesis Summary

Keywords: machine learning, context, reasoning, artificial intelligence, ambient intelligence, sensors

Received: February 2, 2015

This paper presents a summary of the doctoral dissertation of the author, which addresses the task of context-based reasoning in ambient intelligence.

Povzetek: Prispevek predstavlja povzetek doktorske disertacije avtorja, ki obravnava kontekstno sklepanje v ambientalni inteligenci.

1 Introduction

Ambient intelligence (AmI) is a scientific field that refers to environments consisting of smart devices (sensors and actuators) that can sense and respond to the presence of people [1] . The availability of small, wearable, low-cost, power-efficient sensors, combined with advanced signal processing and information extraction, is driving the revolution in AmI domain. This revolution has enabled novel approaches and technologies for accurate measurements in the area of healthcare, enhanced sports and fitness training, and life-style monitoring.

Early AmI systems included a single type of sensors that has made it possible to develop the first proof-of-concept applications. As the field has matured, these systems have gained additional sensors, resulting in the development of advanced and more accurate multi-sensor techniques and applications. However, combining multiple sources of information from multiple sensors is a challenging task. The first issue is that each sensor has its own technical configuration (for example, the data sampling rate) and requires different data-processing techniques in order to first align the different sensor data, and later to extract useful information. The second issue is that even if the multi-source data is aligned, it can be challenging to find an intelligent way to combine this multi-source information in order to reason about the user or the environment. While several approaches for combining multiple sources of information and knowledge have been developed (such as Kalman filters, ensemble learning, and co-training), these approaches have not been specialized for AmI tasks.

The doctoral dissertation [2] addresses the problem of combining multiple sources of information extracted from sensor data by proposing a novel context-based approach called CoReAmI (Context-based Reasoning in Ambient Intelligence). In particular, CoReAmI creates a multi-view perspective, in which each source of information is used as a context separately.

2 The CoReAmI approach

The CoReAmI approach is shown in Figure 1. At the top are the sensors $\{s_1, \ldots , s_m\}$, which provide the raw data. The multiple sensors data is usually represented by multivariate time-series with mixed sampling rates, which are input to CoReAmI. The CoReAmI consists of three phases: (A) context extraction, (B) context modeling and (C) context aggregation.

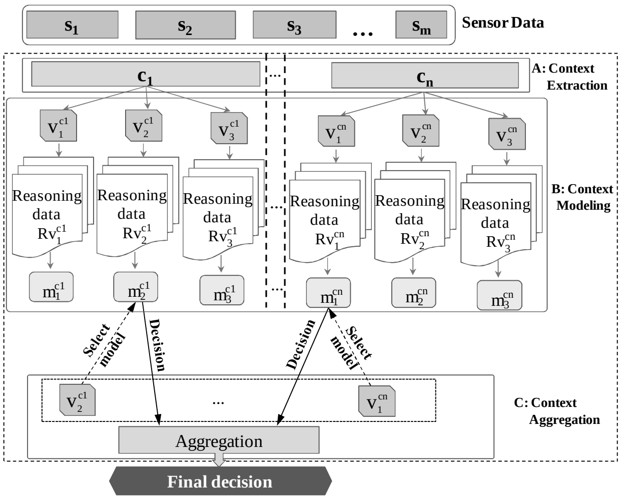

Figure 1. The CoReAmI Approach.

In the first phase (A) the raw sensor data are acquired and the multiple contexts are extracted $\{c_1, \ldots , c_n\}$ using different types of techniques: data-preprocessing techniques, data synchronization, data segmentation, etc. In CoReAmI context represents information about the user which is extracted from the sensor data, e.g., user's activity extracted from wearable accelerometer data. This phase is similar to the feature extraction phase in machine learning (ML). Moreover, the contexts in CoReAmI are features that represent context information. Therefore, each context has values (v^c), which can be numerical or categorical (e.g., "sitting" for the "activity" context).

The second phase B, contains the main logic of the CoReAmI approach. In this phase the context modeling about the problem (activity, fall, energy expenditure, etc.) is performed using the contexts defined in the previous phase. First the context-based partitioning of the dataset is performed, i.e., the dataset is partitioned according to each context and its values. Therefore, for each context value a reasoning model (m^c) is constructed using its reasoning data − the reasoning data is a subset of the whole dataset that has that particular context value (Rv^c). For example, the reasoning data for the "sitting" model will be constructed using the data instances that contain the value "sitting" for the activity context. This way, the approach considers multiple views on the data using each of the features as a context.

In the final phase C, for a given testing data instance, the decisions from each context individually are aggregated and the final decision is provided. In this phase different aggregation techniques can be used, e.g., majority voting, plurality voting, averaging, choosing the median and similar.

The CoReAmI is a general approach for context-based reasoning and can be adapted to a range of tasks in AmI by adapting each of the phases to the particular task.

3 Case studies

The feasibility of the CoReAmI approach was shown in three AmI domains that have emerged as essential building blocks in AmI: activity recognition, energy-expenditure estimation, and fall detection.

The first problem domain, Activity Recognition (AR), can generally be defined as a process of recognizing activities through the analysis of sensor data. In recent years AR gained a lot of research attention, because it provides one of the basic information about a person that is monitored by an AmI system. In our study, we studied the state-of-the-art approaches in AR and observed that it is almost impossible to distinguish standing from sitting activity using a single accelerometer placed on the torso. However, by adapting and applying the CoReAmI approach, we have managed to significantly improve the recognition of these two activities, achieving 86% accuracy − which is for 24 percentage points better than conventional ML approach [4] .

Human Energy-Expenditure (EE) estimation is the process of calculating the amount of expended energy while performing everyday activities. It directly reflects the level of physical activity which makes it important for sports training, weight control, management of metabolic disorders (e.g., diabetes), and other health goals. We adapted and applied the CoReAmI approach to estimate the human energy expenditure using multiple sensor data (accelerometer, heart rate, breath rate, etc.). The CoReAmI significantly improved the estimation performance compared to conventional ML approaches and approaches that are based on single context (such as the activity of the user). Additionally, the CoReAmI provided better energy expenditure estimations than the BodyMedia device, which is a state-of-the-art commercial device for energy expenditure estimation [5] .

The third problem domain on which we applied the CoReAmI approach is the fall detection (FD). FD is a really important application in AmI because falls are among the most critical health problems for the elderly. We adapted and applied the CoReAmI approach to detect human falls. CoReAmI significantly improved the detection performance compared to conventional approaches, such as: threshold-based approaches and approaches based only on ML [6] .

4 Conclusion

This paper summarized the dissertation [2] and presented the main idea and findings of the same. The proposed CoReAmI approach was adapted and tested on three problem domains. The results show that CoReAmI significantly outperforms the competing approaches in each of the domains. This is mainly due to the fact that, by extracting multiple sources of information and combining them by using each source of information as a context, a multi-view perspective is created, which leads to better performance than with conventional approaches.

References

[1] Ducatel K, Bogdanowicz M, Scapolo F, Leijten J, Burgelman J. Scenarios for ambient intelligence in 2010. Technical report. Retrieved September, 2014 from: http://cordis.europa.eu/ist/istagreports.htm

[2] Gjoreski H. Context-based Reasoning in Ambient Intelligence, PhD Thesis, IPS Jožef Stefan, Ljubljana, Slovenia, January, 2015.

[3] Friedman SM, Munoz B, West SK, Rubin GS, Fried LP. Falls and Fear of Falling: Which Comes First? A Longitudinal Prediction Model Suggests Strategies for Primary and Secondary Prevention. Journal of the American Geriatrics Society, 2002; 1329–1335.

[4] Gjoreski H, Kozina S, Luštrek M, Gams M. Using multiple contexts to distinguish standing from sitting with a single accelerometer. European Conference on Artificial Intelligence (ECAI), 2014.

[5] Gjoreski H, Kaluža B, Gams M, Milić R, Luštrek M. Ensembles of multiple sensors for human energy expenditure estimation. In Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing (UbiComp), 2013; 359–362.

[6] Gjoreski H, Gams M, Luštrek M. Context-based fall detection and activity recognition using inertial and location sensors. Journal of Ambient Intelligence and Smart Environments (JAISE), 2014, 6(4); 419–433.

JOŽEF STEFAN INSTITUTE

Jožef Stefan (1835-1893) was one of the most prominent physicists of the 19th century. Born to Slovene parents, he obtained his Ph.D. at Vienna University, where he was later Director of the Physics Institute, Vice-President of the Vienna Academy of Sciences and a member of several scientific institutions in Europe. Stefan explored many areas in hydrodynamics, optics, acoustics, electricity, magnetism and the kinetic theory of gases. Among other things, he originated the law that the total radiation from a black body is proportional to the 4th power of its absolute temperature, known as the Stefan–Boltzmann law.

The Jožef Stefan Institute (JSI) is the leading independent scientific research institution in Slovenia, covering a broad spectrum of fundamental and applied research in the fields of physics, chemistry and biochemistry, electronics and information science, nuclear science technology, energy research and environmental science.

The Jožef Stefan Institute (JSI) is a research organisation for pure and applied research in the natural sciences and technology. Both are closely interconnected in research departments composed of different task teams. Emphasis in basic research is given to the development and education of young scientists, while applied research and development serve for the transfer of advanced knowledge, contributing to the development of the national economy and society in general.

At present the Institute, with a total of about 900 staff, has 700 researchers, about 250 of whom are postgraduates, around 500 of whom have doctorates (Ph.D.), and around 200 of whom have permanent professorships or temporary teaching assignments at the Universities.

In view of its activities and status, the JSI plays the role of a national institute, complementing the role of the universities and bridging the gap between basic science and applications.

Research at the JSI includes the following major fields: physics; chemistry; electronics, informatics and computer sciences; biochemistry; ecology; reactor technology; applied mathematics. Most of the activities are more or less closely connected to information sciences, in particular computer sciences, artificial intelligence, language and speech technologies, computer-aided design, computer architectures, biocybernetics and robotics, computer automation and control, professional electronics, digital communications and networks, and applied mathematics.

The Institute is located in Ljubljana, the capital of the independent state of **Slove**nia (or S♡nia). The capital today is considered a crossroad between East, West and Mediterranean Europe, offering excellent productive capabilities and solid business opportunities, with strong international connections. Ljubljana is connected to important centers such as Prague, Budapest, Vienna, Zagreb, Milan, Rome, Monaco, Nice, Bern and Munich, all within a radius of 600 km.

From the Jožef Stefan Institute, the Technology park “Ljubljana” has been proposed as part of the national strategy for technological development to foster synergies between research and industry, to promote joint ventures between university bodies, research institutes and innovative industry, to act as an incubator for high-tech initiatives and to accelerate the development cycle of innovative products.

Part of the Institute was reorganized into several high-tech units supported by and connected within the Technology park at the Jožef Stefan Institute, established as the beginning of a regional Technology park "Ljubljana". The project was developed at a particularly historical moment, characterized by the process of state reorganisation, privatisation and private initiative. The national Technology Park is a shareholding company hosting an independent venture-capital institution.

The promoters and operational entities of the project are the Republic of Slovenia, Ministry of Higher Education, Science and Technology and the Jožef Stefan Institute. The framework of the operation also includes the University of Ljubljana, the National Institute of Chemistry, the Institute for Electronics and Vacuum Technology and the Institute for Materials and Construction Research among others. In addition, the project is supported by the Ministry of the Economy, the National Chamber of Economy and the City of Ljubljana.

Jožef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
Tel.:+386 1 4773 900, Fax.:+386 1 251 93 85
WWW: http://www.ijs.si
E-mail: matjaz.gams@ijs.si
Public relations: Polona Strnad

INFORMATICA

AN INTERNATIONAL JOURNAL OF COMPUTING AND INFORMATICS

INVITATION, COOPERATION

Submissions and Refereeing

Please submit a manuscript to: http://www.informatica.si/Editors/PaperUpload.asp. At least two referees outside the author's country will examine it, and they are invited to make as many remarks as possible from typing errors to global philosophical disagreements. The chosen editor will send the author the obtained reviews. If the paper is accepted, the editor will also send an email to the managing editor. The executive board will inform the author that the paper has been accepted, and the author will send the paper to the managing editor. The paper will be published within one year of receipt of email with the text in Informatica MS Word format or Informatica LaTeX format and figures in .eps format. Style and examples of papers can be obtained from http://www.informatica.si. Opinions, news, calls for conferences, calls for papers, etc. should be sent directly to the managing editor.

QUESTIONNAIRE

□ Send Informatica free of charge

□ Yes, we subscribe

Please, complete the order form and send it to Dr. Drago Torkar, Informatica, Institut Jožef Stefan, Jamova 39, 1000 Ljubljana, Slovenia. E-mail: drago.torkar@ijs.si

Since 1977, Informatica has been a major Slovenian scientific journal of computing and informatics, including telecommunications, automation and other related areas. In its 16th year (more than twentyone years ago) it became truly international, although it still remains connected to Central Europe. The basic aim of Informatica is to impose intellectual values (science, engineering) in a distributed organisation.

Informatica is a journal primarily covering intelligent systems in the European computer science, informatics and cognitive community; scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the Refereeing Board.

Informatica is free of charge for major scientific, educational and governmental institutions. Others should subscribe (see the last page of Informatica).

ORDER FORM – INFORMATICA

Name: ....

Title and Profession (optional): ....

....

Home Address and Telephone (optional): ....

....

Office Address and Telephone (optional): ....

....

E-mail Address (optional): ....

Signature and Date: ....

Informatica WWW:

http://www.informatica.si/

Referees from 2008 on:

Informatica

An International Journal of Computing and Informatics

Web edition of Informatica may be accessed at: http://www.informatica.si.

Subscription Information Informatica (ISSN 0350-5596) is published four times a year in Spring, Summer, Autumn, and Winter (4 issues per year) by the Slovene Society Informatika, Litostrojska cesta 54, 1000 Ljubljana, Slovenia.
The subscription rate for 2015 (Volume 39) is
– 60 EUR for institutions,
– 30 EUR for individuals, and
– 15 EUR for students
Claims for missing issues will be honored free of charge within six months after the publication date of the issue.

Typesetting: Borut Žnidar.
Printing: ABO grafika d.o.o., Ob železnici 16, 1000 Ljubljana.

Orders may be placed by email (drago.torkar@ijs.si), telephone (+386 1 477 3900) or fax (+386 1 251 93 85). The payment should be made to our bank account no.: 02083-0013014662 at NLB d.d., 1520 Ljubljana, Trg republike 2, Slovenija, IBAN no.: SI56020830013014662, SWIFT Code: LJBASI2X.

Informatica is published by Slovene Society Informatika (president Niko Schlamberger) in cooperation with the following societies (and contact persons):
Robotics Society of Slovenia (Jadran Lenarčič)
Slovene Society for Pattern Recognition (Janez Perš)
Slovenian Artificial Intelligence Society (Dunja Mladenić)
Cognitive Science Society (Urban Kordeš)
Slovenian Society of Mathematicians, Physicists and Astronomers (Andrej Likar)
Automatic Control Society of Slovenia (Sašo Blažič)
Slovenian Association of Technical and Natural Sciences / Engineering Academy of Slovenia (Vojteh Leskovšek)
ACM Slovenia (Andrej Brodnik)

Informatica is financially supported by the Slovenian research agency from the Call for co-financing of scientific periodical publications.

Informatica is surveyed by: ACM Digital Library, Citeseer, COBISS, Compendex, Computer & Information Systems Abstracts, Computer Database, Computer Science Index, Current Mathematical Publications, DBLP Computer Science Bibliography, Directory of Open Access Journals, InfoTrac OneFile, Inspec, Linguistic and Language Behaviour Abstracts, Mathematical Reviews, MatSciNet, MatSci on SilverPlatter, Scopus, Zentralblatt Math

Volume 39 Number 1 March 2015 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

Informatica **39** (2015) Number 1, pp. 1–101