Web-Based Lemmatisation of Named Entities

Richárd Farkas¹, Veronika Vincze², István Nagy², Róbert Ormándi¹,
György Szarvas¹, and Attila Almási²

MTA-SZTE, Research Group on Artificial Intelligence,
6720 Szeged, Aradi Vértanúk tere 1., Hungary,
University of Szeged, Department of Informatics,
6720 Szeged, Árpád tér 2., Hungary
{rfarkas, vinczev, ormandi, szarvas}@inf.u-szeged.hu,
Istvan.Nagy@stud.u-szeged.hu, vizipal@gmail.com

Abstract. Identifying the lemma of a Named Entity is important for many Natural Language Processing applications like Information Retrieval. Here we introduce a novel approach for Named Entity lemmatisation which utilises the occurrence frequencies of each possible lemma. We constructed four corpora in English and Hungarian and trained machine learning methods using them to obtain simple decision rules based on the web frequencies of the lemmas. In experiments our web-based heuristic achieved an average accuracy of nearly 91%.

Keywords: Lemmatisation, web-based techniques, named entity recognition.

1 Introduction

This paper seeks to lemmatise Named Entities (NEs) in English and in Hungarian. Finding the lemma (and inflectional affixes) of a proper noun can be useful for several reasons: the proper name can be stored in a normalised form (e.g. for indexing) and it may prove to be easier to classify a proper name in the corresponding NE category using its lemma instead of the affixed form. Next, the inflectional affixes themselves can contain useful information for some specific tasks and thus can be used as features for classification (e.g. the plural form of an organisation name is indicative of org-for-product metonymies and is a strong feature for proper name metonymy resolution [1]). It is useful to identify the phrase boundary of a certain NE to be able to cut off inflections. This is not a trivial task in free word order languages where NEs of the same type can follow each other without a punctuation mark.

The problem is difficult to solve because, unlike common nouns, NEs cannot be listed. Consider, for instance, the following problematic cases of finding the lemma and the affix of a NE:

1. the NE ends in an apparent suffix,
2. two (or more) NEs of the same type follow each other and they are not separated by punctuation marks,
3. one NE contains punctuation marks within.

In morphologically rich languages such as Hungarian, nouns (hence NEs as well) can have hundreds of different forms owing to grammatical number, possession marking and grammatical cases. When looking for the lemmas of NEs (case 1), the word form being investigated is deprived of all the suffixes it may bear. However, there are some NEs that end in an apparent suffix (such as *McDonald's* or *Philips* in English or *Pannon* in Hungarian, with *-on* meaning “on”), but this pseudo-suffix belongs to the lemma of the NE and should not be removed. Such proper names make the lemmatisation task non-trivial.

In the second and third cases, when two NEs of the same type follow each other, they are usually separated by a punctuation mark (e.g. a comma). Thus, if present, the punctuation mark signals the boundary between the two NEs. However, the assumption that punctuation marks are constant markers of boundaries between consecutive NEs and that the absence of punctuation marks indicates a single (longer) name phrase often fails, and thus a more sophisticated solution is necessary to locate NE phrase boundaries. Counterexamples for this naive assumption are NEs such as *Stratford-upon-Avon*, where two hyphens occur within one single NE.

In order to be able to select the appropriate lemma for each problematic NE, we applied the following strategy. Each ending that seems to be a possible suffix is cut off the NE. With those NEs that consist of several tokens either separated by punctuation marks or not, every possible cut is performed. Then Google and Yahoo searches are carried out on all the possible lemmas. We trained several machine learning classifiers to find the decision boundary for appropriate lemmas based on the frequency results of the search engines.

2 Related Work

Lemmatisation of common nouns. Lemmatisation, that is, dividing the word form into its root and suffixes, is not a trivial issue in morphologically rich languages. In agglutinative languages such as Hungarian, a noun can have hundreds of different forms owing to grammatical number, possession marking and a score of grammatical cases: e.g. a Hungarian noun has 268 different possible forms [2]. On the other hand, there are lemmas that end in an apparent suffix (which is obviously part of the lemma), thus sometimes it is not clear what belongs to the lemma and what functions as a suffix. However, the lemmatisation of common nouns can be made easier by relying on a good dictionary of lemmas [3].

Lemmatisation of NEs. The problem of proper name lemmatisation is more complicated since NEs cannot be listed exhaustively, unlike common nouns, due to their diversity and increasing number. Moreover, NEs can consist of several tokens, in contrast to common nouns, and the whole phrase must be taken into account. Lots of suffixes can be added to them in Hungarian (e.g. *Invitelben*, where *Invitel* is the lemma and *-ben* means “in”), and they can bear the plural or genitive marker *-s* or *'s* in English (e.g. *Toyotas*). What is more, there are NEs that end in an apparent suffix (such as *McDonald's*), but this pseudo-suffix belongs to the lemma of the NE.

NE lemmatisation has not attracted much attention so far because it is not such a serious problem in major languages like English and Spanish as it is in agglutinative

languages. An expert rule-based and several string distance-based methods for Polish person name inflection removal were introduced in [4]. And a corpus-based rule induction method was studied for every kind of unknown word in Slovene in [5]. The scope of our study lies between these two as we deal with different kinds of NEs. On the other hand, these studies focused on removing inflection suffixes, while our approach handles the separation of consecutive NEs as well.

Web-based techniques. Our main hypothesis is that the lemma of an NE has a relatively high frequency on the World Wide Web, in contrast to the frequency of the affixed form of the NE. Although there are several papers that tell us how to use the WWW to solve simple natural language problems like [6,7], we think that this will become a rapidly growing area and more research will be carried out over the next couple of years. To the best of our knowledge, there is currently no procedure for NE lemmatisation which uses the Web as a basis for decision making.

3 Web-Based Lemmatisation

The use of online knowledge sources in Human Language Technology (HLT) and Data Mining tasks has become an active field of research over the past few years. This trend has been boosted by several special and interesting properties of the World Wide Web. First of all, it provides a practically limitless source of (unlabeled) data to exploit and, more importantly, it can bring some dynamism to applications. As online data change and rapidly expand over time, a system can remain up-to-date and extend its knowledge without the need for fine tuning or any human intervention (like retraining on up-to-date data). These features make the Web a very useful source of knowledge for HLT applications as well. On the other hand, accessing the data is feasible just via search engines (e.g. we cannot iterate through all of the occurrences of a word). There are two interesting problems here: first, appropriate queries must be sent to a search engine; second, the response of the engine offers several opportunities (result frequencies, snippets, etc.) in addition to simply “reading” the pages found.

In the case of lemmatisation we make use of the result frequencies of the search engines. In order to be able to select the appropriate lemma for each problematic NE, we applied the following strategy. In step-by-step fashion, each ending that seems to be a possible suffix is cut off the NE. As for the apparently multiple NEs separated by punctuation marks or not, every possible cut is performed; that is, in the case of *Stratford-upon-Avon*, the possibilities *Stratford* + *upon* + *Avon* (3 lemmas), *Stratford-upon* + *Avon*, *Stratford* + *upon-Avon* (2 lemmas), *Stratford-upon-Avon* (1 lemma) are generated. Then after having found all the possible lemmas we do a Google search and a Yahoo search. Our key hypothesis is that the frequency of the lemma-candidates on the WWW is high – or at least the ratio of the full form and lemma-candidate frequencies is relatively high – with an appropriate lemma and low in incorrect cases.

In order to verify our hypothesis we manually constructed four corpora of positive and negative examples for NE lemmatisation. Then, Google and Yahoo searches were performed on every possible lemma and training datasets were compiled for machine learning algorithms using the queried frequencies as features. The final decision rules were learnt on these datasets.

4 Experiments

In this section we describe how we constructed our datasets and then investigate the performance of several machine learning methods in the tasks outlined above.

4.1 The Corpora

The lists of negative and positive examples for lemmatisation were collected manually. We adopted the principal rule that we had to work on real-world examples (we did not generate fictitious examples), so our three annotators were asked to browse the Internet and collect “interesting” cases. These corpora are the unions of the lists collected by 3 linguists and were checked by the chief annotator. The samples mainly consist of person names, company names and geographical locations occurrences on web pages. Table 1 lists the size of each corpora (constructed for the three problems). The corpora are accessible and are free of charge.

The first problematic case of finding the lemma of a NE is when the NE ends in an apparent suffix (which we will call the *suffix* problem in the following). In agglutinative languages such as Hungarian, NEs can have hundreds of different inflections. In English, nouns can only bear the plural or the possessive marker *-s* or *'s*. There are NEs that end in an apparent suffix (such as *Adidas* in English), but this pseudo-suffix belongs to the lemma of the NE.

We decided to build two corpora for the suffix problem; one for Hungarian and one for English and we produced the possible suffix lists for the two languages. In Hungarian more than one suffix can be matched to several phrases. In these cases we examined every possible cut and the correct lemma (chosen by a linguist expert) became a positive example, while every other cut was treated as a negative one.

The other two lemmatisation tasks, namely the case where several NEs of the same type follow each other and they are not separated by punctuation marks and the case where one NE contains punctuation marks within are handled together in the following (and is called the *separation* task). When two NEs of the same type follow each other, they are usually separated by a punctuation mark (e.g. a comma or a hyphen). Thus, if present, the punctuation mark signals the boundary between the two NEs (e.g. *Arsenal-Manchester final*; *Obama-Clinton debate*; *Budapest-Bécs marathon*). However, the assumption that punctuation marks are constant markers of boundaries between consecutive NEs and that the absence of punctuation marks indicates a single (longer) name phrase often fails, and thus a more sophisticated procedure is necessary to locate NE phrase boundaries. Counterexamples for this naive assumption are NEs such as the *Saxon-Coburg-Gotha family*, where the hyphens occur within the NE, and sentences such as *Gyurcsány Orbán gazdaságpolitikájáról mondott véleményyt.* (*'Gyurcsány expressed his views on Orbán's economic policy.'* (two consecutive entities) as opposed to *“He expressed his views on Orbán Gyurcsány's economic policy.”* (one single two-token-long entity)). Without background knowledge of the participants in the present-day political sphere in Hungary, the separation of the above two NEs would be a pose problem. Actually, the first rendition of the Hungarian sentence conveys the true, intended meaning; that is, the two NEs are correctly separated. As for the second version, the NEs are not separated and are treated as a two-token-long entity. In Hungarian,

Table 1. The sizes of the corpora

	Suffix Eng	Suffix Hun	Separation Eng	Separation Hun
positive examples	74	207	51	137
negative examples	84	543	34	69

however, a phrase like *Gyurcsány Orbán* could be a perfect full name, *Gyurcsány* being a family name and *Orbán* being – in this case – the first name.

As consecutive NEs without punctuation marks appear frequently in Hungarian due to the free word-order, we decided to construct a corpus of negative and positive cases for Hungarian. Still, such cases can occur just as the consequence of a spelling error in English. Hence we focused on special punctuation marks which can separate entities in several cases (*Obama-Clinton debate*) but are part of the entity in others. In the separation task there are several cases where more than one cut is possible (more than two tokens in Hungarian and more than one special mark in English). In such cases we again asked a linguist expert to choose the correct cut and every incorrect cut became a negative example.

4.2 The Feature Set

To create training datasets for machine learning methods – which try to learn how to separate correct and incorrect cuts based on labeled examples – we sent queries to the Google and Yahoo search engines using their APIs¹. The queries started and finished in quotation marks and the *site:.hu* constraint was used in the Hungarian part of the experiments. In the suffix tasks, we sent queries with and without suffixes to both engines and collected the number of hits. The original database contained four dimensional feature vectors. Two dimensions listed the number of Google hits and two components listed similar values from the Yahoo search engine.

The original training datasets for the separation tasks contained six features. Two stood for the number of Google hits for the potential first and second parts of a cut and one represented the number of Google hits for the whole observed phrase. The remaining three features conveyed the same information, but here we used the Yahoo search engine. Each of the four tasks was a binary classification problem. The class value was set to 0 for negative examples and 1 for positive ones.

Our preliminary experiments showed that using just the original form of the datasets (Non-Rate rows of Table 2) is not optimal in terms of classification accuracy. Hence we performed some basic transformations on the original data, the first component of the feature vector was divided by the second component. If the given second component was zero, then the new feature value was also zero (Rate rows in Table 2). This yielded a two dimensional dataset when we utilised both Yahoo and Google hits for the suffix classification tasks and a four dimensional for the separation tasks. Finally, we took the minimum and the maximum of the separated parts' ratios hence providing the possibility to learn rules such as "*if **one of the** separated parts' frequency ratio is higher than X*".

¹ Google API: <http://code.google.com/apis/soapsearch/>; Yahoo API: <http://developer.yahoo.com/search/>

4.3 Machine Learning Methods

In the tests, we experimented with several machine learning algorithms. Each method was evaluated using the 10-fold-cross-validation method. Here classification accuracy was used as the evaluation metric in each experiment. We employed the WEKA [8] machine learning library to train our models. Different kinds of learning algorithms and parameter settings were examined to achieve the highest possible accuracy. The first one is a baseline method which is the naive classifier; it classifies each sample using the most frequent label observed on the training dataset.

We used the k-Nearest Neighbour [9] algorithm, which is one of the simplest machine learning algorithms. In this algorithm, an object is classified by taking the majority vote of its neighbours, with the object being assigned to the class most common amongst its k nearest neighbours. Here k is a positive integer, and it is typically not very large.

We also used the C4.5 [10] decision tree learning algorithm as it is a widely applied method in data mining. In a decision tree each interior node corresponds to an attribute; an edge to a child represents a possible value or an interval of values of that variable. A leaf represents the most probable class label given the values of the variables represented by the path from the root. A tree can be learned by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner. The recursion stops either when further splitting is not possible or when the same classification can be applied to each element of the derived subset. The M parameter of the C4.5 defines the minimum number of instances per leaf, i.e. there are no more splits on nodes if the number of instances is fewer than M.

Logistic Regression [11] is another well studied machine learning method. It seeks to maximise the conditional probability of classes, subject to feature constraints (observations). This is performed by weighting features so as to maximise the likelihood of data.

The results obtained by using the different feature sets and the machine learning algorithms described above are presented in Table 2 for the suffix task, and in Table 3 for the separation task.

4.4 Discussion

For the English suffix task, the size of the databases is quite small, which leads to a dominance of the k-Nearest Neighbour classifier, since the lazy learners – like k-Nearest Neighbour – usually achieve good results on small datasets.

In this task the training algorithms attain their best performance on the transformed datasets using rates of query hits (this holds when Yahoo or Google searches were performed). One could say that the rate of the hits (one feature) is the best characterisation in this task. However, we can see that with the Hungarian suffix problem the original dataset characterises the problem better, and thus the transformation is really unnecessary.

The best results for the Hungarian suffix problem are achieved on the full dataset, but they are almost the same as those for untransformed Yahoo dataset. Without doubt, this is due to the special property of the Yahoo search engine which searches accent sensitively, in contrast to Google. For example, for the query *Ottó* Google finds every webpage which contains *Ottó* and *Otto* as well, while Yahoo just returns the *Ottó*-s.

Table 2. Suffix task results obtained from applying different learning methods. (The first letter stands for the language /E – English; H – Hungarian/. The second letter stands for the search engine used /B – Both; G – Google; Y – Yahoo/. NR means Non-Rate database, R means Rate database. Thus for example, H-B-NR means Hungarian problem using non-rate dataset and both search engines).

	kNN k=3	kNN k=5	C4.5 M=2	C4.5 M=5	Log Reg	Base
E-B-NR	89.24	89.24	86.71	84.18	84.81	53.16
E-B-R	93.04	89.87	91.77	92.41	73.42	53.16
E-G-NR	87.34	86.71	87.34	81.65	82.28	53.16
E-G-R	93.67	93.67	87.97	92.41	90.51	53.16
E-Y-NR	89.87	89.87	86.08	85.44	84.18	53.16
E-Y-R	91.77	91.77	87.34	91.77	88.61	53.16
H-B-NR	94.27	94.27	82.67	90.00	88.27	72.40
H-B-R	84.67	86.13	81.73	81.73	72.40	72.40
H-G-NR	85.33	85.33	82.40	82.93	83.33	72.40
H-G-R	83.60	78.40	83.60	77.60	77.60	72.40
H-Y-NR	93.73	93.73	83.87	88.13	86.13	72.40
H-Y-R	87.20	84.27	87.20	74.00	74.00	72.40

Table 3. Separation task results obtained from applying different learning methods

	kNN k=3	kNN k=5	C4.5 M=2	C4.5 M=5	Log Reg	Base
English	88.23	84.71	95.29	94.12	77.65	60.00
Hungarian	79.25	81.13	80.66	79.72	70.31	64.63

The separation problem for Hungarian proved to be a difficult task. The decision tree (which we found to be the best solution) is a one-level high tree with a split. This can be interpreted as *if one of the resulting parts' frequency ratio is high enough, then it is an appropriate cut*. It is interesting to see that among the learned rules for the English separation task, there is a constraint for the second part of a possible separation (while the learnt hypothesis for Hungarian consisted of simple *if (any) one of the resulting parts is ... rules*).

5 Conclusions

In this paper we introduced corpora for the English and Hungarian Named Entity lemmatisation tasks. The corpora are freely available for further comparison studies. The NE lemmatisation task is very important for textual data indexing systems, for instance, and is of great importance for agglutinative (Finno-Ugric) languages and other languages that are rich in inflections (like Slavic languages). To handle this task, we proposed a web-based heuristic which sends queries for every possible lemma to two popular web search engines. Based on the constructed corpora we automatically derived simple decision rules on the search engine responses by applying machine learning methods.

Despite the small size of our corpora, we got fairly good empirical results and even better accuracies can probably be obtained with bigger corpora. The 91.09% average accuracy score supports our hypothesis that even the frequencies of possible lemmas provide enough information to help find the right separations. We encountered several problems when using search engines to obtain the lemma-frequencies like, the need of an accent sensitive search, difficulties in the proper handling of punctuation marks in queries and that the frequency values of results are estimated values only. We think that if offline corpora with an appropriate size were available, the frequency counts would be more precise and our heuristic could probably attain even better results.

Acknowledgments. This work was supported in part by the NKTH grant of Jedlik Ányos R&D Programme 2007 of the Hungarian government (codename TUDORKA7). The author wishes to thank the anonymous reviewers for valuable comments.

References

1. Farkas, R., Simon, E., Szarvas, Gy., Varga, D.: Gyder: Maxent metonymy resolution. In: Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007), Prague, Czech Republic, pp. 161–164. Association for Computational Linguistics (2007)
2. Melčuk, I.: Modèle de la déclinaison hongroise. In: Cours de morphologie générale (théorique et descriptive), Montréal, Les Presses de l'Université de Montréal, CNRS (edn). vol. 5, pp. 191–261 (2000)
3. Halácsy, P., Trón, V.: Benefits of resource-based stemming in Hungarian information retrieval. In: Peters, C., Clough, P., Gey, F.C., Karlgren, J., Magnini, B., Oard, D.W., de Rijke, M., Stempfhuber, M. (eds.) CLEF 2006. LNCS, vol. 4730, pp. 99–106. Springer, Heidelberg (2007)
4. Piskorski, J., Sydow, M., Kupść, A.: Lemmatization of polish person names. In: Proceedings of the Workshop on Balto-Slavonic Natural Language Processing, Prague, Czech Republic, pp. 27–34. Association for Computational Linguistics (2007)
5. Erjavec, T., Dzeroski, S.: Machine learning of morphosyntactic structure: Lemmatizing unknown Slovene words. *Applied Artificial Intelligence* 18, 17–41 (2004)
6. Bunesco, R.C., Pasca, M.: Using encyclopedic knowledge for named entity disambiguation. In: EACL (2006)
7. Etzioni, O., Cafarella, M., Downey, D., Popescu, A.M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Unsupervised named-entity extraction from the web: an experimental study. *Artif. Intell.* 165, 91–134 (2005)
8. Witten, I.H., Frank, E.: *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd edn. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, San Francisco (2005)
9. Aha, D.W., Kibler, D., Albert, M.K.: Instance-based learning algorithms. *Mach. Learn.* 6, 37–66 (1991)
10. Quinlan, J.R.: *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco (1993)
11. Berger, A.L., Pietra, S.D., Pietra, V.J.D.: A maximum entropy approach to natural language processing. *Computational Linguistics* 22, 39–71 (1996)