Clustering Semantically Equivalent Words into Cognate Sets in Multilingual Lists

Bradley Hauer and Grzegorz Kondrak
Department of Computing Science
University of Alberta
Edmonton, Alberta, Canada, T6G 2E8
{bmhauer,gkondrak}@ualberta.ca

Abstract

Word lists have become available for most of the world's languages, but only a small fraction of such lists contain cognate information. We present a machine-learning approach that automatically clusters words in multilingual word lists into cognate sets. Our method incorporates a number of diverse word similarity measures and features that encode the degree of affinity between pairs of languages. The output of the classification algorithm is then used to generate cognate groups. The results of the experiments on word lists representing several language families demonstrate the utility of the proposed approach.

1 Introduction

Cognates are words with a shared linguistic origin, such as English *father* and French *père*. Identification of cognates is essential in historical linguistics, and cognate information has been successfully applied to natural language processing tasks, such as sentence alignment in bitexts (Simard et al., 1993), and statistical machine translation (Kondrak et al., 2003). The problem of automatically identifying pairs of cognates has been addressed previously (Frunza and Inkpen, 2009; Kondrak, 2009). The process of identification is usually based on the combination of the following three types of evidence: phonetic similarity, semantic similarity, and recurrent sound correspondences. The input data include dictionaries, multilingual word lists, and bitexts. The objective can be finding pairs of cognates among two related languages, or finding groups of cognates among multiple languages.

In this paper, we focus on the task of identifying cognate groups (clusters) in word lists on the basis of word similarity and language relationships. Word lists are now available for most of the world's languages (Wichmann et al., 2011). However, only a fraction of such lists contain cognate information. Methods proposed for pairwise cognate identification are of limited utility for such data because they fail to consider the transitivity property of the cognation relationship. Cognate groups are also more useful than cognate pairs as the input to algorithms for reconstructing phylogenetic trees (Bouchard-Côté et al., 2007).

A number of word similarity measures have been applied to the problem of cognate identification (Frunza et al., 2005). Kondrak and Dorr (2004) report that a simple average of various measures outperforms any individual measure of phonetic similarity. We propose to combine measures using a machine-learning approach based on support vector machines (SVMs). The SVMs are trained on both positive and negative examples, and allow for a seamless combination of a number of diverse similarity measures.

In addition to word similarity features, we include a set of language-pair features that incorporate information regarding the degree of relatedness between languages. We develop a way to self-train these features in the absence of preexisting cognate information. We also present a novel clustering algorithm for defining cognate groups, which utilizes the classification decisions generated by the SVM classifier. We evaluate our method on two sets of word lists representing several language families. The results demonstrate the utility of the proposed techniques.

This paper is structured as follows. In Section 2, we define the task that we address in this work. In Section 3, we discuss relevant previous work. In Section 4, we describe our method of clustering cognates. Section 5 explains our evaluation methodology. Sections 6 and 7 report the results of our experiments. We conclude with a discussion of our work, its implications, and the potential for further research.

Proceedings of the 5th International Joint Conference on Natural Language Processing, pages 865–873,
Chiang Mai, Thailand, November 8 – 13, 2011.

2 Problem definition

There are over five thousand languages in the world, which are grouped into dozens of language families (Lewis, 2009). Some language families, such as Indo-European and Austronesian, are very well documented. There are many languages, however, for which the only available data are relatively short vocabulary lists. For example, most languages in the Automated Similarity Judgement Program database (Wichmann et al., 2011) are represented by word list composed of only 40 meanings. Other lists of the most stable meanings range from 15 to 200 (Dyen et al., 1992).

	ALL	AND	ANIMAL	...
Irish	uile	agus	ainhme	...
Welsh	pob	a	anifail	...
Breton	holl	hag	aneval	...
Rumanian	toti	iar	animal	...
Italian	tutto	ed	animale	...
...	...	...	...	...

Table 1: A sample of the Indo-European Database used in our experiments.

Table 1 visualizes a small part of the typical dataset as a two-dimensional $m \times n$ table, in which rows represent n individual languages and columns represent m distinct meanings. The meanings are limited to the basic vocabulary that is relatively resistant to lexical replacement, and present in most of the world's languages (Swadesh, 1952). The task that we address in this paper is the identification of cognate groups (clusters) within each column. The number of clusters can range between 1 and n. Since many datasets contain either orthographic forms or use an approximate phonetic encoding, we do not require the words to be fully phonetically transcribed. If the data contains multiple words per language/meaning slot, we randomly pick one of the forms and discard the others. We will evaluate our methods by comparing the generated clusters to cognate judgements made by linguists that are experts in language families represented by the datasets.

3 Previous work

Frunza et al. (2005) experiment with several Weka classifiers (Hall et al., 2009) that combine various orthographic similarity measures for the pairwise identification of cognates and false friends[1] as aids to second-language learners. Datasets were extracted from various sources: a manually aligned bitext, lists of cognates and false friends, and exercises for language learners. No single classifier is reported as the most accurate on all tasks. Our approach differs in the focus on identifying cognate groups for the purpose of classifying related languages.

Mulloni (2007) applies the SVMTool tagger (Giménez and Màrquez, 2004) to automatically generate words in one language from their cognates in a related language. He reports a 30-35% accuracy on an English-German cognate list. A relatively large list of cognates (1683 entries) was used for training the SVMTool. This underlines the inherent problem with the proposed methods: in order to identify or generate cognates between languages, a substantial number of cognates must have already been identified. We are interested in a more realistic scenario where *no* cognate pairs are available.

Bouchard-Côté et al. (2007) present a unified stochastic model of diachronic phonology aimed at the automatic reconstruction of proto-forms and deriving phylogenetic trees. They assume the cognate groups to be the input to their model. In the actual experiments on four closely-related Romance languages they filtered out non-cognates by thresholding the normalized edit distance scores. They consider the joint modelling of phonology with the determination of cognates as "an interesting direction for future work." We include edit distance as one of the features in our SVM model.

Hall and Klein (2010) point out the limitations of pairwise cognate identification, and present a generative phylogenetic model for determining cognate groups from unaligned word lists. However, their method requires the language family tree to be known beforehand. This is a difficult prerequisite to satisfy as phylogenetic trees are rarely uncontroversial even in the case of well-studied families. In their experiments, they also disregard the semantic information, instead applying their method to randomly scrambled cognate groups from three closely related Romance languages. In contrast, our experimental setup emulates a much more realistic scenario.

[1] False friends are words that are orthographically similar but historically unrelated, such as English *dinner* and Spanish *dinero*.

4 Clustering cognates

We propose a discriminative approach to clustering cognates. We start by formulating cognate identification as a binary classification task on pairs of words that have the same meaning in different languages. Our model is trained on annotated data from a subset of the dataset, and applied to a different, disjoint subset. The classification results are then used to cluster words into cognate sets. In this section we discuss various features used for training the binary classifier, as well as the details of our clustering approach.

The principal idea for cognate identification follows from the observation that, on average, cognate pairs display higher word similarity that non-cognates. Since no single word similarity measure may be sufficient, we want to utilize a combination of measures. We opt for a feature-based approach because it is more principled and flexible than a simple average or a linear combination of scores. It also allows us to seamlessly incorporate a set of language-pair features that provide additional context for the classification.

We considered various software packages, including Weka (Hall et al., 2009), SVM-light (Joachims, 1999), Liblinear (Fan et al., 2008), and LibSVM (Chang and Lin, 2011). We ultimately selected LibSVM for our experiments due to its higher overall performance in development.

4.1 Word similarity features

We selected the following word similarity features on the basis of the results of our preliminary development experiments:

- minimum edit distance
- the longest common prefix length
- number of common bigrams
- the length of each word (2 separate features)
- the difference in length between the longer and the shorter word

During development, we also experimented with other features, but decided not to include them in the final system for various reasons. The longest common subsequence length was considered redundant with edit distance; longest common substring length and the number of common trigrams were mostly subsumed by bigrams; and shared first letter was generalized by the prefix length.

We decided to exclude features based on phonetic similarity in order to ensure the applicability to datasets that contain only orthographic forms.

4.2 Language-pair features

A limitation of the word similarity features is their strictly local application to pairs of words. However, it is useful to consider not only the words, but also the languages they come from. Intuitively, if we observe that two language lists contain a large number of similar word pairs, we would expect the languages to be closely related, and therefore share many cognates. Conversely, if there is little overall similarity, we may suspect that that cognates will be rare or non-existent.

As an example, consider the average value of the normalized minimum edit distance computed between semantically equivalent words across pairs of word lists. For French and Italian, which are closely related Romance languages, the value is 0.44. In contrast, for French and German, which are more remotely related, the value is only 0.18. Indeed, among 199 word pairs each, there are 155 cognate pairs between French and Italian, and only 48 between French and German. The similarity between cognates is also expected to be greater between strongly related languages, as there would, on average, have been less divergence due to the more recent linguistic split. Once again, our example supports this idea: the average similarity between French/Italian cognates is 0.52, while the average similarity between French/German cognates is 0.22. We would like our classifier to take advantage of this tendency.

Our solution is to introduce a set of binary language pair features, one for each pair of languages in the data. For example, any instance consisting of a German word and an English word has a feature corresponding to that language pair set to '1', and all other language-pair features set to '0'. The language-pair features elevate the learned classification model from a local model to a global one, allowing it to make connections between languages, rather than words alone. For example, if relatively many training instances that have the German-English feature set to '1' are cognate, this information is expected to increase cognate recognition accuracy for this language pair at test time.

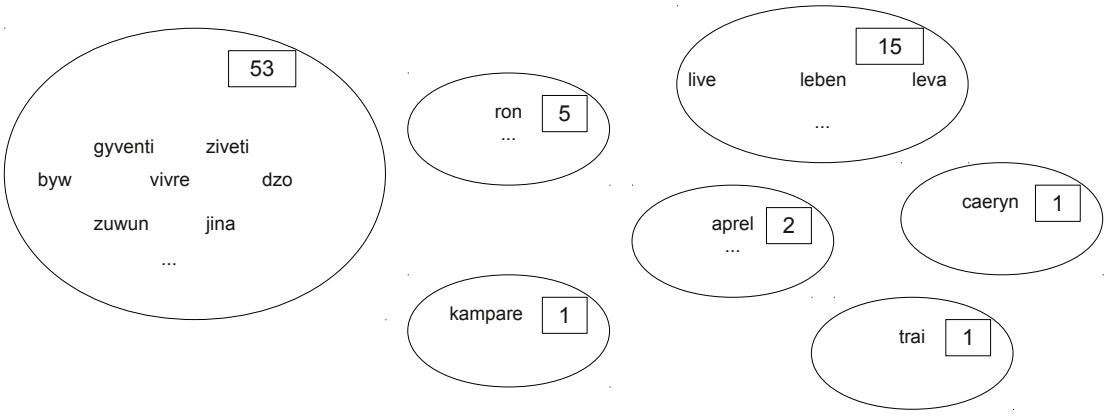

Figure 1: A cognate clustering of of the words with the meaning "to live". The boxed numbers are the total number of words in that cluster.

4.3 Self-training the language-pair features

As with any features, the weights for language-pair features must be derived from annotated training data. For example, in order to derive a useful weight for the German-English feature, we need training instances consisting of English and German words, which are annotated either as cognates or as non-cognates. However, in a realistic scenario of our approach being applied to a previously unanalyzed family of languages, no cognate information may be available. Thus we are faced with a circular problem: language-pair features are likely to improve cognate identification accuracy, but we need to have at least some cognates already identified in order to train the language-pair features.

Our proposed solution to the problem is to train the weights of the language-pair features on our own classifications. We adopt a two-pass approach. We first train a model and classify the data without the language-pair features; language information from the training data is not used in any way. These initial classifications of the test data are then treated as correct, thus providing a 'best guess' at what the cognate classifications would be, were they available. Next, we re-train the model using the initial classification, but this time utilizing the language-pair features. The second and final classification is obtained with the new model, which is expected to be more accurate. This method allows the language-pair features to be used to good effect on any set of languages, using only information locally available in the data to be classified.

4.4 Clustering

After the pairwise cognate classification is completed, the final task is the formation of cognate groups, or clusters. An example of such a clustering is shown in Figure 1, wherein words from various languages for the meaning "to live" have been correctly placed in cognate groups. Each ellipse is a cluster; two words are cognate if and only if they are in the same cluster. Obtaining clusterings such as these for arbitrary data is one objective of our research.

Because of the transitivity property of cognates, there is often no clustering that is completely consistent with the pairwise classification decisions. Each triple of words x, y, and z involves three binary classification instances: $x - y$, $y - z$, and $x - z$. The contradiction arises if two instances are classified as positive, and the remaining one is classified as negative. Consider, for example, the words *beva*, *bivi*, and *vivir*, all of which mean "to live" in Breton, Sardinian, and Spanish, respectively. It is reasonable to expect that *beva* and *bivi* will be identified as cognate, as will *bivi* and *vivir*, due to the similarity between these two pairs. The remaining pair, *beva* and *vivir*, have much lower similarity, and could reasonably be identified as non-cognate. In fact, all three of these words are cognate with each other. A clustering approach would first put two of the words into a single cluster; the remaining word would be added due to its similarity with one of the first two words. Thus, even though the cognate relationship between *beva* and *vivir* may not be found directly,

the links *beva-bivi* and *bivi-vivir* are enough; our clustering method finds a cognate pair that would otherwise have been overlooked.

In order to come up with a clustering, we typically need to override some of the binary classifications. Blindly following the transitivity property is likely to result in clusters that are excessively large, since some positive classifications are caused by accidental similarities between non-cognate words. Consider another pair of words with the meaning "to live": Breton *beva* and Swedish *leva*. While obviously similar (differing by only a single letter substitution), they are, in fact, not cognate. This presents a challenge: finding a proper additional condition that should be satisfied before merging clusters, in order to avoid such accidental merges.

The solution employed by our clustering method is based on the notion of *average similarity* between clusters. Initially, each word corresponding to a particular meaning is placed in its own cluster. We then consider each word pair that has been classified as cognate, and for each pair decide whether the corresponding clusters should be merged. We compute the average value of cognate judgements between the two clusters, which is a measure of how similar the two clusters are. If this value is less than a certain threshold (optimized during development), the merge is aborted, and clustering continues as it would had the two words been judged non-cognate. Otherwise, the clusters containing each word are merged.

5 Evaluation of clustering quality

Pairwise classification is typically evaluated by some combination of the following four well-known measures: accuracy (correct classifications divided by all classifications), precision (true positives divided by all positive classifications), recall (true positives divided by actual positives), and F-score (the harmonic average of precision and recall). However, evaluating clustering is a more complex problem than evaluating pairwise classifications. Pairwise metrics have significant weaknesses when applied to clustering, where a word is proposed to be cognate with all words in its cluster. Incorrectly assigning a word to a large cluster will generate a large number of false positives, while incorrectly assigning a word to a smaller cluster would generate fewer false positives. On the other hand, incorrectly positing two clusters instead of one is penalized proportionally to the square of the size of the cluster. In short, the number of false positives and false negatives an error creates may not be balanced or consistent.

We considered a number of alternative metrics, eventually deciding on the B-Cubed measure. B-Cubed metrics assign a precision and recall to each item in a set of clusters — in our case, to each word. The item precision is the ratio of the number of its cognates in its cluster to the number of items in its cluster. The item recall is the ratio of the number of cognates in its cluster to the total number of its cognates. A B-Cubed F-score is computed from the B-Cubed precision and recall, analogously to pairwise F-score. Amigó et al. (2009) show that B-Cubed metrics satisfy four constraints deemed critical to a valid clustering evaluation metric, while all other metrics investigated, including pairwise metrics, fail at least one of these criteria.

For an illustrative example, consider again the words shown in Figure 1. The gold standard indicates one large cluster (53 words), one medium sized cluster (15 words), and several smaller clusters. In this case, incorrectly assigning (or failing to assign) a single word to the large cluster produces many false positives (or false negatives), while exchanging a word between two smaller clusters has a minimal effect. In order to verify the inconsistency of pairwise metrics on clustering, we analyzed two different clusterings of these words; one was an excessively aggressive clustering, in which the medium-sized cluster was almost entirely subsumed into the larger cluster. The other was a more conservative clustering which formed the medium-sized cluster comparatively well, and generally better recovered the overall structure of the actual clusters. The pairwise metrics reported the more aggressive clustering to be significantly better than the more conservative result; the B-Cubed metrics did not display this anomalous behavior.

The above example demonstrates that pairwise metrics can, under realistic conditions, report a much worse clustering to be significantly better, demonstrating their intrinsic volatility: a single error can have dramatic, unpredictable effects that depend more on chance similarities than on the quality of the clustering or classification process. B-Cubed metrics do not have this problem; being an average of item-based measurements, er-

rors will have consistent effects, balanced against the resulting quality of the clusters. We therefore adopt B-Cubed metrics as the preferred measure of clustering performance.

6 The Indo-European experiments

Our first experiment involves an extremely well-studied family, for which we also have access to relatively long and complete lists of basic words. We divide the data into training and test sets along different sets of meanings. The same languages appear in both sets.

6.1 Data

The publicly-available Comparative Indo-European Database (Dyen et al., 1992) contains words for 200 different meanings in 95 languages of the Indo-European family. The words in each meaning are grouped in cognate sets. We used a pre-processed version of the data which places each meaning in an individual file, for a total of 200 files. Each word is labelled with the number of the cognate set that it belongs to.[2] We randomly selected 20 out of 200 meanings data as a development set. We also created a separate held-out test set of 20 meanings, roughly 10% of the data. We performed two tests: one with a small 20-meaning training set, and the other with a large 180-meaning training set, which also included the development set. In both cases, the test set was the same. We made sure that the sets of meanings in the training and test data were disjoint.

6.2 Classification methods

We tested three methods of classification, each making different use of language-pair features (LPF):

- NO LPF: a strictly "local" method that considers only the pair of words in question, and utilizes no language-pair features.
- SUPERVISED LPF: the weights of language-pair features are trained on the annotated instances in the training set.
- SELF-TRAINED LPF: the two-pass approach described in section 4.3. No cognate information for the language pairs in the test set is assumed to be in the training set.

[2]This processed data is available on request.

Method	Size of Training Set	
	20	180
Baseline	0.623	0.623
NO LPF	0.642	0.687
SELF-TRAINED LPF	0.656	0.687
SUPERVISED LPF	0.677	0.677

Table 2: Average B-Cubed F-Scores for the Indo-European data.

For the baseline, we adopt a simple but surprisingly effective method of grouping words according to their first letter or phoneme. Two cognates maintaining their common initial sound are assigned to the same cluster by this baseline. However, unrelated words that accidentally share the same first letter are also marked as cognate. In the example shown in Figure 1, the baseline correctly identifies the middle-sized cluster of words from 15 Germanic languages, but splits the largest cluster into several smaller ones, containing words starting with b, d, g, j, and z, respectively.

6.3 Results

Table 2 shows the results in terms of average B-Cubed F-score. Our methods consistently outperform the first-letter baseline regardless of the size of the training set. When the 20 meaning training subset was used, the results rank the SELF-TRAINED LPF method between the SUPERVISED LPF and NO LPF methods. When the 180 meaning training set was used, the NO LPF and SELF-TRAINED LPF models achieved substantial improvement voer the baseline; however, somewhat surprisingly, the SUPERVISED LPF model obtained a smaller improvement. This suggests that the SUPERVISED LPF model may have issues with overspecialization, or may not be able to make use of data past a certain point. The improvement exhibited by the SELF-TRAINED LPF model suggest that the two-pass method makes greater use of additional training data, and is capable of producing even better clusters than the SUPERVISED LPF model.

7 The ASJP experiments

This set of experiments involved some of the relatively short lists from a comprehensive database that contains most of the world's languages. This time, we divided the data into the training and test sets by languages, rather than by meanings.

Baseline	0.653
NO LPF	0.662
SELF-TRAINED LPF	0.714
SUPERVISED LPF	0.703

Table 3: Average B-Cubed F-Scores for ASJP data, with languages grouped by family.

Baseline	0.660
NO LPF	0.724
SELF-TRAINED LPF	0.701
SUPERVISED LPF	0.665

Table 4: Average B-Cubed F-Scores for ASJP data, with languages ordered alphabetically.

7.1 Data

Our second dataset consists of word lists from the Automated Similarity Judgement Program (ASJP) project, which represent 92 languages belonging to 5 language families: Austro-Asiatic, Hmong-Mien, Mixe-Zoque, Sino-Tibetan, and Tai-Kadai. Each list contains 40 basic meanings transcribed in a phonetic notation devised for the ASJP.

We performed two experiments on this data. In each experiment, the first 46 languages were used for training, and the other 46 were used for testing. For the first experiments, the languages were grouped according to language families, ensuring that most families appeared exclusively either in the training or the test set (one group was split between the two sets). For the second test, we sorted the languages alphabetically, essentially shuffling different language families. One of our objectives was to determine how the distribution of the languages affects the results. The first experiment adopts natural divisions between language families, emulating a realistic scenario where a model is trained on well-studied families and applied to the data representing less-studied families. The second experiment represents the situation where we have annotation for some of the languages in a family, and aim to discover cognates among other languages in the same family. The alphabetic ordering is akin to random shuffling of languages, but has the advantage of being easy to replicate.

7.2 Results on the data grouped by family

In this experiment, languages were naturally arranged into language families. That is, all languages in the same language family, such as Sino-Tibetan and Tai-Kadai, are adjacent in the data. This provides a realistic testing environment, wherein one set of language families (with known cognate data) is used to obtain cognate information for another set of language families.

Table 3 shows that the SELF-TRAINED LPF model obtains the best results in this experiment. The model appears able to generalize well, though it is not trained on any of the language pairs it is tested on. The NO LPF method and the baseline method are less effective. The SUPERVISED LPF method is again less effective that the SELF-TRAINED LPF method. This is likely caused by the lack of training data that contains the information for the language pairs that it encounters in the test set.

7.3 Results on the data sorted alphabetically

This experiment is based on the same ASJP dataset, with the exception that the languages are sorted alphabetically. This strengthens the relationship between the training and test sets (as languages from all families can be found in each), while reducing the similarities within them.

The results of this test are presented in Table 4. The NO LPF approach was a surprise winner this time. It appears that the presence of more linguistic relationships between the training and testing sets was the deciding factor. We conjecture that the relationship between the training and test sets was sufficiently strong for a locally trained model to provide very good results.

On the other hand, the SUPERVISED LPF approach performed much worse, only barely exceeding the baseline results. This likely occurs for the same reason that the NO LPF method does better — there are not enough similarities between the training and the test data to use global features accurately.

Notably, the SELF-TRAINED LPF approach still does quite well, even with fewer similarities to use. This demonstrates high reliability, as all other methods do poorly on at least one test; only the SELF-TRAINED LPF method provides high quality clusterings throughout all experiments.[3]

8 Discussion

Based on our results, we are able to conclude that our methods can consistently and accurately

[3]We also considered training on the Indo-European data and testing on ASJP data, but unfortunately the two datasets use entirely different notations: Romanized orthography versus specialized phonetic encoding. No simple method exists for converting between the two.

identify and cluster cognates, exceeding the performance of a strong baseline method. We have shown how SVMs can produce accurate cognate classifications; we have also shown how these classifications can be systematically used to create cognate groups. Evaluated with the B-Cubed metrics, these clusters are demonstrated to be of high quality compared to known cognate groups.

Language pairs appear to be most useful as binary features in a supervised SVM model if the data to be classified are from a small number of language families. In cases where there are fewer similarities in the data, fewer connections between languages can be learned. Relying on the cognate annotations substantially lowers the quality of the results in such situations, as the data it requires is not available. We thus find the SUPERVISED LPF method not only impractical, but also rather unreliable.

In contrast, our SELF-TRAINED LPF approach has been demonstrated to be a reliable, high-performing method, which produces good classifications in all of our experiments, throughout which the size of the training data and the distribution of the languages to be classified varies significantly. While not always yielding the best clusters overall, the SELF-TRAINED LPF method has been shown to consistently yield very good results across all tests. It functions well under realistic conditions, as it does not require cognate information on the languages to be classified. Furthermore, it can find global connections between languages that do not have cognate information available. We thus take our results as a recommendation for the use of the SELF-TRAINED LPF approach, and for the further investigation of language-pair features in cognate identification in general.

9 Conclusion

We have proposed an effective new method of identifying cognates that can make useful global connections from local data. Our demonstration that SVMs can make use of language information to improve cognate classifications lays a foundation for the use of cognate judgements in language classification and provides insight into how machine learning methods can be used successfully for the purposes of cognate identification.

Further work in this area might process language pairs directly using a method similar to our own, developing a machine learning, cognate based method for language classification. Brown et al. (2008) notes that cognate judgements could be used to compare and classify languages, but that this is yet to be done. Our use of relationships between language pairs to assist in classification sets a strong precedent for cognate-based language classification. In addition, other machine learning algorithms, such as Bayesian classifiers, as well as sophisticated phonetic similarity measures, may produce more accurate cognate classifications and clusters, and could be tested in future studies.

Acknowledgements

We thank Eric Holman, Søren Wichmann, and other members of the ASJP project for sharing their cognate-annotated data sets. We also thank Shane Bergsma for insightful comments. Format conversion of the Comparative Indo-European Database was performed by Qing Dou. This research was partially funded by the Natural Sciences and Engineering Research Council of Canada.

References

Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdego. 2009. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, pages 461–486.

Alexandre Bouchard-Côté, Percy Liang, Thomas Griffiths, and Dan Klein. 2007. A probabilistic approach to diachronic phonology. *Proceedings of the 2007 Conference on Empirical Methods on Natural Language Processing (EMNLP07)*.

Cecil H. Brown, Eric W. Holman, Søren Wichmann, and Viveka Velopillai. 2008. Automated classification of the World's languages: A description of the method and preliminary results. *Language Typology and Universals 61:4*, pages 285–308.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Isidore Dyen, Joseph Kruskal, and Paul Black. 1992. An indoeuropean classication: A lexicostatistical experiment. *Transactions of the American Philosophical Society*.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Oana Frunza and Diana Inkpen. 2009. Identification and disambiguation of cognates,false friends, and partial cognates using machine learning techniques. *International Journal of Linguistics,Vol. 1, No. 1*.

Oana Frunza, Diana Inkpen, and David Nadeau. 2005. A text processing tool for the romanian language. *Proceedings of the EuroLAN 2005 Workshop on Cross-Language Knowledge Induction*.

Jesús Giménez and Lluís Màrquez. 2004. Svmtool: A general pos tagger generator based on support vector machines. *Journal of Machine Learning Research*.

David Hall and Dan Klein. 2010. Finding cognate groups using phylogenies. *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1030–1039.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: An update. *SIGKDD Explorations, Volume 11, Issue 1*.

T. Joachims. 1999. Making large-scale svm learning practical. *Advances in Kernel Methods - Support Vector Learning*.

Grzegorz Kondrak and Bonnie J. Dorr. 2004. Identification of confusable drug names: A new approach and evaluation methodology. *Proceedings of the Twentieth International Conference on Computational Linguistics (COLING 2004)*, pages 952–958.

Grzegorz Kondrak, Daniel Marcu, and Kevin Knight. 2003. Cognates can improve statistical translation models. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 46–48.

Grzegorz Kondrak. 2009. Identification of cognates and recurrent sound correspondences in word lists. *TAL Volume 50*, pages 201–235.

M. Paul Lewis. 2009. *Ethnologue: Languages of the World, 16th edition*. Dallas, Tex.: SIL International.

Andrea Mulloni. 2007. Automatic prediction of cognate orthography using support vector machines. *Meeting of the ACL: Student Research Workshop, 2007*.

Michel Simard, George F. Foster, and Pierre Isabelle. 1993. Using cognates to align sentences in bilingual corpora. *CASCON '93 Proceedings of the 1993 conference of the Centre for Advanced Studies on Collaborative research: distributed computing - Volume 2*.

Morris Swadesh. 1952. Lexico-statistic dating of prehistoric ethnic contacts. *Proceedings of the American philosophical society*, 96:452–463.

Soren Wichmann, Andre Muller, Viveka Velupillai, Annkathrin Wett, Cecil H. Brown, Zarina Molochieva, Sebastian Sauppe, Eric W. Holman, Pamela Brown, Julia Bishoffberger, Dik Bakker, Johann-Mattis List, Dmitry Egorov, Oleg Belyaev, Matthias Urban, Robert Mailhammer, Helen Geyer, David Beck, Evgenia Korovina, Pattie Epps, Pilar Valenzuela, Anthony Grant, and Harald Hammarstrom. 2011. The ASJP Database (version 14). `http://email.eva.mpg.de/~wichmann/ASJPHomePage.htm`.