TKK Reports in Information and Computer Science

Espoo 2010 TKK-ICS-R37

PROCEEDINGS OF THE MORPHO CHALLENGE 2010 WORKSHOP

Mikko Kurimo, Sami Virpioja and Ville T. Turunen (Editors)

Aalto University
School of Science
and Technology

TKK Reports in Information and Computer Science

Espoo 2010 TKK-ICS-R37

PROCEEDINGS OF THE MORPHO CHALLENGE 2010 WORKSHOP

Mikko Kurimo, Sami Virpioja and Ville T. Turunen (Editors)

Aalto University School of Science and Technology
Faculty of Information and Natural Sciences
Department of Information and Computer Science

Aalto-yliopiston teknillinen korkeakoulu
Informaatio- ja luonnontieteiden tiedekunta
Tietojenkäsittelytieteen laitos

Distribution:
Aalto University School of Science and Technology
Faculty of Information and Natural Sciences
Department of Information and Computer Science
PO Box 15400
FI-00076 AALTO
FINLAND
URL: http://ics.tkk.fi
Tel. +358 9 470 01
Fax +358 9 470 23369
E-mail: series@ics.tkk.fi

ISBN 978-952-60-3330-3 (Online)
ISSN 1797-5042 (Online)
URL: http://lib.tkk.fi/Reports/2010/isbn9789526033303.pdf

AALTO ICS
Espoo 2010

Abstract: In natural language processing many practical tasks, such as speech recognition, information retrieval and machine translation depend on a large vocabulary and statistical language models. For morphologically rich languages, such as Finnish and Turkish, the construction of a vocabulary and language models that have a sufficient coverage is particularly difficult, because of the huge amount of different word forms. In Morpho Challenge 2010 unsupervised and semi-supervised algorithms are suggested to provide morpheme analyses for words in different languages and evaluated in various practical applications. As a research theme, unsupervised morphological analysis has received wide attention in conferences and scientific journals focused on computational linguistic and its applications. This is the proceedings of the Morpho Challenge 2010 Workshop that contains one introduction article with a description of the tasks, evaluation and results and six articles describing the participating unsupervised and supervised learning algorithms. The Morpho Challenge 2010 Workshop was held at Espoo, Finland in 2-3 September, 2010.

Keywords: morpheme analysis, unsupervised learning, semisupervised learning, information retrieval, statistical machine translation

Acknowledgement: This work was supported by the Academy of Finland in the project *Adaptive Informatics*, Graduate School of Language Technology in Finland, and in part by the IST Programme of the European Community, under the FP7 project EMIME (213845) and the Challenge Program of PASCAL2 Network of Excellence.

CONTENTS

Overview and Results of Morpho Challenge 2010

Mikko Kurimo, Sami Virpioja and Ville T. Turunen
Adaptive Informatics Research Centre, Aalto University School of Science and Technology
P.O.Box 15400, FIN-00076 Aalto, Finland
Mikko.Kurimo@tkk.fi

Abstract

In Morpho Challenge 2010 unsupervised and semi-supervised algorithms are suggested to provide morpheme analyses for words in different languages that can be used in various practical applications. Morpheme analysis is particularly useful in speech recognition, information retrieval and machine translation for morphologically rich languages where the amount of different word forms is very large. The evaluations in Morpho Challenge consist of: 1. comparisons to grammatical morphemes, 2. information retrieval experiments based on morphemes instead of words, and 3. machine translation experiments where morpheme and word based systems are combined. The evaluation languages are: Finnish, Turkish, German, and English. This overview paper describes the goals, data, tasks, participants, evaluations, and obtained results. The Morpho Challenge is part of the EU Network of Excellence PASCAL2 Challenge Program.

1 Introduction

A common task in applications such as speech recognition, machine translation and information retrieval is to construct a vocabulary and a statistical language model for all words that will be used. For many languages, particularly the morphologically rich ones, the vast amount of various inflected forms in which the words appear poses an important challenge. By using a rule-based morphological analyzer that exist already for quite many languages, most word forms can be returned to their base forms. However, these analyzers do not cover the whole language and leave out many word forms that are either rare, foreign, or ungrammatical. While the frequency of the unanalyzed types maybe small in running text or speech, they might still be meaningful for the application. A special challenge is posed by less resourced languages for which the available morphological analyzers are particularly poor.

The series of the Morpho Challenge competitions, started in 2005, has supported the research for unsupervised morpheme analysis by providing annual evaluations for shared tasks using shared training data for various languages. The goal has been to develop unsupervised and language independent machine learning algorithms that could discover morphemes from large amount of given raw text data. The algorithms have been evaluated, not only by comparing the obtained morphemes to the linguistic ones, but by testing them in real NLP applications using state-of-the-art technology. The achieved results show which algorithms produce the most useful morpheme analysis and how useful they are compared to rule-based systems and other state-of-the-art solutions in the task.

In the Morpho Challenge 2010, both unsupervised and semi-supervised learning algorithms have been evaluated. The semi-supervised learning is allowed to utilize the provided small amount of labeled data to enhance or classify the analysis obtained by unsupervised learning from the large unlabeled data. Alternatively, the core analyses obtained first from the limited labeled data can be extended by using the large unlabeled data. For Morpho Challenge the semi-supervised task has become particularly interesting, because small samples of morphologically labeled words can already be obtained for many languages. From the point of view of the machine learning research, the semi-supervised learning task in Morpho Challenge is an opportunity to develop and evaluate methods for morphology modeling and more generally, sequence segmentation and labeling, which have not yet been extensively studied.

The evaluation tasks in 2010 are essentially the same as in 2009: comparison to grammatical morphemes, information retrieval from text and machine translation. All the analysis languages in 2010 were also used in 2009: Finnish, Turkish, German, and English. However, the amount of training data was substantially increased by including all the information retrieval and machine translation data into the shared word list. This simplified the comparison of all methods in all tasks, but made the learning of the morpheme analysis more data intensive.

2 Data and tasks

2.1 Data sets

The word list for each language has been constructed by collecting word forms occurring in text corpora. The text corpora have been obtained by combining collections from the Wortschatz collection[1] at the University of Leipzig, CLEF[2], and the Europarl corpus (Koehn, 2005). The total size is 18.8 million sentences for English, 6.6 million for Finnish, 9.7 million for German, and 1 million for Turkish. The corpora have been preprocessed for the Morpho Challenge (tokenized, lower-cased, some conversion of character encodings).

For strictly unsupervised learning, only the word list were to be used. For semi-supervised training, the desired correct analyses, based on linguistic gold standards, were supplied for a random sample of 1000 words per each language. The participants could use either the gold standard segmentations (available only for English, Finnish, and Turkish) or directly the gold standard labels (for any language). In addition, independent development sets of circa 700 words were provided. They could be used for obtaining a rough estimate of the performance and tuning some parameters of the learning method. If either the training or the development sets were not needed by the participants, they could use a combined set as a larger training or development set. Neither the training sets nor the development sets contained any of the word forms in the final test sets of Competition 1.

2.2 Competition 1

In Competition 1, for each language, the proposed morpheme analyses are compared against a linguistic gold standard. Since the learning task is unsupervised or semi-supervised, it cannot be expected that the algorithm comes up with morpheme labels that exactly correspond to the ones designed by linguists. That is, no direct comparison will take place between the labels in the proposed analyses and the labels in the gold standard.

What can be expected, however, is that two word forms that contain the same morpheme according to the proposed analysis also have a morpheme in common according to the gold standard. For instance, in the English gold standard, the words “foot” and “feet” both contain the morpheme “foot_N”. It is thus desirable that also the applied algorithm discovers a morpheme that occurs in both these word forms, be it called “FOOT”, “morpheme784”, “foot” or something else.

The gold standard reference analyses were the same as in the Morpho Challenges 2007–2009 (Kurimo et al., 2008; Kurimo et al., 2009; Kurimo et al., 2010b) and the evaluation method is the same as in Morpho Challenge 2009 (Kurimo et al., 2010b). The test sets included 10000 (English), 200000 (Finnish), 50000 (German), and 50000 (Turkish) random words from the gold standards.

In practice, the evaluation is done by sampling a large number of word pairs, such that both words in the pair have at least one morpheme in common. Then the *precision* is calculated as the proportion of morpheme sharing word pairs in the participant’s sample that really has a morpheme in common according to the gold standard. Correspondingly, the *recall* is calculated as the proportion of morpheme sharing word pairs in the gold standard sample that also exist in the participant’s submission. As the evaluation measure, we will use *F-measure*, which is the harmonic mean of precision and recall:

$$\text{F-measure} = 1/(1/\text{Precision} + 1/\text{Recall})\,. \quad (1)$$

As exemplified by Spiegler and Monson (2010), the evaluation method applied in Morpho Challenges has some drawbacks. Most importantly, it is prone to artificial boosting of recall by giving several alternative analyses per word or adding a unique shared morpheme for the analysis of each word. Spiegler and Monson (2010) propose a new evaluation method, EMMA, which applies a graph-based assignment algorithm. In EMMA,

[1] http://corpora.informatik.uni-leipzig.de/

[2] http://www.clef-campaign.org/

each proposed morpheme is matched for each morpheme in the gold standard. This matching allows direct calculation of precision (how many morphemes in the proposed analysis are in the gold standard analysis) and recall (how many morphemes in the gold standard analysis are in the proposed analysis) for each word.

Although the official results of Competition 1 are calculated with the Morpho Challenge 2009 metric, we have evaluated all the submissions also with EMMA and give the results of both evaluations in Section 4. Due to the computational complexity of the EMMA method, the full test sets applied in the standard evaluation could not be used. Instead, the evaluation is performed by sampling 10 random subsets of 1000 words from the test sets and calculating the scores for each subset.

2.3 Competition 2

In Competition 2, the submitted morpheme analyses were evaluated by using them in an Information Retrieval (IR) task. IR evaluation corpora were available for English, German and Finnish and were provided by the Cross-Language Evaluation Forum (CLEF). To evaluate the algorithms, the IR experiments were run after replacing all word forms in the corpora and the queries by the submitted analyses. Morpheme analysis is important in IR, since the user will want to retrieve all relevant documents irrespective of which word forms are used to describe the contents. Especially, for highly inflective language like Finnish, retrieval performance will be very bad if morphology is not taken into account. Traditionally, language dependent rule based methods like two-level morphological analysis or stemming are used. These type of methods were also tested for comparison.

The task, the data and the evaluation tools are the same as used in Morpho Challenge 2009 (Kurimo et al., 2010b). The Lemur Toolkit (Ogilvie and Callan, 2002) with Okapi BM25 ranking was used. The evaluation criterion was Mean Average Precision (MAP). For each submission, a stop list was generated, since Okapi BM25 suffers greatly if the corpus contains terms that are very common. The morpheme segmentation algorithms introduce such terms when they e.g. separate suffixes. Any term that has a collection frequency higher than 75000 (Finnish) or 150000 (German and English) was added to the stop list and thus excluded from indexing. In this year's challenge, all three tasks used the same combined word list. While constructing the new word lists, some small preprocessing issues were fixed. All reference method experiments were repeated with the new word lists and the effect of the changes was found to be minimal. The results are comparable to the results of previous challenges.

The performance of the participating algorithms was compared to a number of reference methods, some of which are commonly used in IR. The reference methods are the same as used in Morpho Challenge 2009 (Kurimo et al., 2010b).

1. *Morfessor Categories-MAP*: The Morfessor Categories-MAP was used for the unsupervised morpheme analysis. The stem vs. suffix tags were kept, but did not receive any special treatment in the indexing.

2. *Morfessor Baseline*: Morfessor Baseline algorithm was used to split words into smaller pieces without any real morpheme analysis.

3. *dummy*: No segmentation or analysis was performed and words were used as index terms as such. Hyphens were replaced by spaces so that hyphenated words were indexed as separate words.

4. *Grammatical*: The words were analyzed using the same gold standard analyses in each language that were utilized as the "ground truth" in the Competition 1. Either only the first interpretation was used ("Grammatical First") or all of them ("Grammatical All"). Words that were not in the gold standard were indexed as such. Because our gold standards are quite small, 60k (English) - 600k (Finnish), compared to the amount of words that the unsupervised methods can analyze, we did not expect "Grammatical" to perform particularly well.

5. *snowball*: No real morpheme analysis was performed, but the words were stemmed by language specific stemming algorithms provided by Snowball libstemmer library [3]. Hyphenated words were first split to parts that were then stemmed separately.

[3] http://snowball.tartarus.org/

6. *TWOL*: Two-level morphological analyzer TWOL from Lingsoft Inc.[4] was used to find the normalized forms of the words which were then used as index terms. Some words may have several alternative interpretations and either all alternatives were used ("TWOL all") or only the first one ("TWOL first"). Compound words were split to parts. Words not recognized by the analyzer were indexed as such.

7. *Best Prev.*: This is the algorithm in each task that provided the highest average precision in previous Morpho Challenges.

2.4 Competition 3

In Competition 3, the morpheme analyses proposed by the algorithms were evaluated in a statistical machine translation (SMT) framework. The two source languages used in the competition were Finnish and German. In principle, the evaluation is simple: We train a translation system that can translate the morphologically analyzed Finnish or German sentence to English. Then, we use it to translate new sentences, and compare the translation results to the reference translations. If the morphological analysis is good, it reduces the sparsity of the data and helps the translation task. If the analysis contains many errors, it degrades the translation results.

The basic setup is similar to the one proposed for Finnish-to-English translation by de Gispert et al. (2009): The translation models were trained to translate from a morphologically complex source language to English. The words of the source language were replaced by their morpheme analyses before training the translation models. The morpheme-based models are combined to a standard word-based model by generating n-best lists of translation hypotheses from both models, and finding the best overall translation with the Minimum Bayes Risk (MBR) decoding (Kumar and Byrne, 2004).

The final SMT systems were evaluated by measuring the similarity of the translation results to a human-made reference translation using the BLEU metric (Papineni et al., 2002). BLEU is based on the co-occurrence of n-grams: It counts how many n-grams (for $n = 1, \ldots, 4$) the proposed translation has in common with the reference translations and calculates a score based on this. Although BLEU is a very simple measure, it usually corresponds well to human evaluations if the compared translation systems are similar.

While the basic setup in the evaluation is the same as in Morpho Challenge 2009 (Kurimo et al., 2010b), one change was made to make the competition more fair: As the alignment tool used in training the SMT system has a limitation of 100 tokens per sentence, we discarded all sentences that had more than 100 letters. This way, all systems had, in practice, the same amount of training data.

For training and testing the SMT systems, the Europarl data sets were divided into three subsets: training set for training the models, development set for tuning the model parameters, and test set for evaluating the translations. For the Finnish-English systems, we had 325 561 sentences for training, 2 849 for tuning, and 3 000 for testing. For the German-English systems, we had 317 137 sentences for training, 2 665 for tuning, and 3 000 for testing. Note that the word lists given for participants for learning morphology included all the word forms in the Europarl corpus.

3 Submitted algorithms

Research groups from four different universities and institutions submitted the results of their algorithms. The authors and the names of their algorithms are listed in Table 1. Sample analyses from the submissions (100 words per language) are available from the Morpho Challenge web pages[5].

Statistics of the output of the submitted algorithms are presented in Tables 2–5 for each of the languages. The column "Type" shows how the methods exploit the provided labeled data sets: "S" denotes a semi-supervised algorithm, "P" denotes an unsupervised algorithm with supervised parameter tuning (i.e., only the development sets were used), and "U" denotes a fully unsupervised algorithm (i.e., only the word lists were used). The average amount of analyses per word is shown in the column "#a/w" and the average amount of morphemes per analysis in the column "#m/w". Only the DEAP algorithm provided alternative analyses. The total amount of morpheme types is given in the column "#lexicon".

As baseline results for unsupervised morpheme analysis, the organizers provided mor-

[4] http://www.lingsoft.fi/

[5] http://www.cis.hut.fi/morphochallenge2010/samples/

Table 1: The participants and the names of their algorithms.

Author	Affiliation	Algorithm name
Golénia et al.	University of Bristol, UK	MAGIP
Kohonen et al.	Aalto University, FI	Morfessor S+W
Kohonen et al.	Aalto University, FI	Morfessor S+W+L
Kohonen et al.	Aalto University, FI	Morfessor U+W
Lignos	University of Pennsylvania, USA	Base Inference
Lignos	University of Pennsylvania, USA	Aggressive Compounding
Lignos	University of Pennsylvania, USA	Iterative Compounding
Nicolas et al.	UNSA + CNRS, FR & Univ. de A Coruña, ES	MorphAcq
Spiegler et al.	University of Bristol, UK	DEAP MDL-CAT
Spiegler et al.	University of Bristol, UK	DEAP MDL-NOCAT
Spiegler et al.	University of Bristol, UK	DEAP PROB-CAT
Spiegler et al.	University of Bristol, UK	DEAP PROB-NOCAT
Spiegler et al.	University of Bristol, UK	Promodes
Spiegler et al.	University of Bristol, UK	Promodes-H
Spiegler et al.	University of Bristol, UK	Promodes-E

pheme analysis by a publicly available unsupervised algorithm called “Morfessor Categories-MAP”developed at Helsinki University of Technology (Creutz and Lagus, 2005a). Analysis by the original Morfessor Baseline (Creutz and Lagus, 2002; Creutz and Lagus, 2005b), which provides only a surface-level segmentation, was also provided for reference. Additionally, the reference results were provided for “letters”, where the words are simply split into letters.

4 Results

4.1 Competition 1

Tables 6, 7, 8 and 9 show the results of the linguistic evaluation with the Morpho Challenge metric. The best result is obtained by a different algorithm for each of the languages: Morfessor S+W for English, DEAP MDL-NOCAT for Finnish, Morfessor U+W for German, and Morfessor S+W+L for Turkish. Overall, strong performance is measured for the semi-supervised Morfessor algorithms by Kohonen et al., semi-supervised DEAP algorithms by Spiegler et al., and all unsupervised algorithms by Lignos.

The results are somewhat different with our alternative evaluation method, EMMA, as shown in Tables 10, 11, 12 and 13. The most evident difference is that the DEAP submissions that included alternative analyses for the words, do not perform as well. Also the algorithms that applied parameter optimization with the Morpho Challenge evaluation, such as Morfessor U+W, often lose their positions. The best overall results are obtained by Morfessor S+W+L by Kohonen et al., which gets top score for every language for which it was evaluated. For German, unsupervised MorphAcq by Nicolas et al. is the best among the submissions, but the reference method Morfessor Categories-MAP gets even higher scores. All three unsupervised algorithms Lignos are again strong, but now Base Inference is clearly outperforms the other two.

4.2 Competition 2

Tables 14, 15 and 16 show the obtained MAP values for the submissions in English, Finnish and German respectively. For English and Finnish, Base Inference and Aggressive Compounding methods by Lignos gave the best performance respectively and for German Morfessor U+W by Kohonen et al. Overall, these algorithms, especially Aggressive Compounding, gave good results for all languages. The best performance for all three languages, was achieved by one of the reference algorithms. The rule based word normalizer, TWOL, gave best performance in German and Finnish. In the English task, TWOL was only narrowly beaten by the traditional Porter stemmer.

Compared to previous years, the best result for Finnish was very slightly improved, but the Paramor+Morfessor algorithm from 2008 (Monson et al., 2009) was still unbeaten for English and German. However, the comparison is not com-

Table 2: Statistics of the analyses for **English** words. Type indicates the level of supervision used in the method, #a/w is the average amount of analyses per word, #m/w is the average amount of morphemes per analysis, and #lexicon is the size of the morph lexicon.

Author	Method	Type	#a/w	#m/w	#lexicon
Golénia et al.	MAGIP	S	1.00	3.87	173204
Kohonen et al.	Morfessor S+W	S	1.00	2.80	33759
Kohonen et al.	Morfessor S+W+L	S	1.00	2.70	52323
Kohonen et al.	Morfessor U+W	P	1.00	2.96	14677
Lignos	Aggressive Compounding	U	1.00	2.30	275340
Lignos	Base Inference	U	1.00	2.11	302911
Lignos	Iterative Compounding	U	1.00	2.10	309887
Nicolas et al.	MorphAcq	U	1.00	1.51	617531
Spiegler et al.	DEAP MDL-CAT	S	4.29	3.86	780699
Spiegler et al.	DEAP MDL-NOCAT	S	1.76	3.88	414247
Spiegler et al.	DEAP PROB-CAT	S	4.71	3.23	1459539
Spiegler et al.	DEAP PROB-NOCAT	S	1.76	3.38	557190
Spiegler et al.	Promodes	P	1.00	4.21	119854
Spiegler et al.	Promodes-E	P	1.00	3.51	185814
Spiegler et al.	Promodes-H	P	1.00	4.91	71610
-	Morfessor Baseline	U	1.00	2.30	72752
-	Morfessor Categories-MAP	U	1.00	2.34	256686
-	letters	-	1.00	9.83	64

pletely fair since Paramor+Morfessor was a combination of two separate algorithms, Paramor and Morfessor. Combining any two methods from this year would likely offer better performance than either of the methods alone.

Statistical testing was performed by transforming the obtained Average Precision values with the $f(x) = \arcsin(\sqrt{x})$ function to make them more normally distributed. Significances were computed using Two-way ANOVA and 95% confidence level. For more details, see Morpho Challenge 2009 overview (Kurimo et al., 2010b). The "top group" or the submissions that have no significant difference to the best result of each language are marked with X (Tables 14-16).

4.3 Competition 3

In Competition 3, we calculated the BLEU scores both for the individual systems, including a word-based system, and for MBR combination with the word-based system. The statistical significances ($p < 5\%$) of the results were inspected with Wilcoxon signed-rank test on ten subsets of the test data. In addition to the submitted algorithms, we have the reference Morfessor methods, word-based baseline, and grammatical morphemes based on the gold standards. The gold standards did not include analyses for all the words in the data; unknown word forms were left unprocessed. In the case of several alternative analyses, we selected the one with the least number of morphemes.

Tables 14 and 15 show the machine translation results for Finnish and German, respectively. In Finnish translation, the best individual system was based on Morfessor Baseline, outperforming also word-based translation. Base Inference by Lignos was the best among the submissions. Also Morfessor Categories-MAP, grammatical morphemes, Iterative Compounding by Lignos and Morfessor U+W by Kohonen et al. were not significantly worse than the word-based model. With MBR combination, Morfessor Baseline was still the best. The only one not significantly worse than it was another Morfessor method, Categories-MAP. Also grammatical morphemes, Base Inference and Iterative Compounding by Lignos, and Morfessor S+W by Kohonen et al., and DEAP MDL-CAT by Spiegler et al. could significantly improve the results of the word-based model.

In German translation, the best individual system was again based on Morfessor Baseline. The best among the submitted methods was Morfes-

Table 3: Statistics of the analyses for **Finnish** words. Type indicates the level of supervision used in the method, #a/w is the average amount of analyses per word, #m/w is the average amount of morphemes per analysis, and #lexicon is the size of the morph lexicon.

Author	Method	Type	#a/w	#m/w	#lexicon
Golénia et al.	MAGIP	S	1.00	4.43	983445
Kohonen et al.	Morfessor S+W	S	1.00	4.27	14133
Kohonen et al.	Morfessor S+W+L	S	1.00	4.34	20589
Kohonen et al.	Morfessor U+W	P	1.00	4.07	3716
Lignos	Aggressive Compounding	U	1.00	3.49	450668
Lignos	Base Inference	U	1.00	2.64	674204
Lignos	Iterative Compounding	U	1.00	2.80	738489
Spiegler et al.	DEAP MDL-CAT	S	5.33	3.22	3374816
Spiegler et al.	DEAP MDL-NOCAT	S	3.29	3.41	2320673
Spiegler et al.	DEAP PROB-CAT	S	5.34	2.17	9739221
Spiegler et al.	DEAP PROB-NOCAT	S	2.36	2.30	4287514
Spiegler et al.	Promodes	P	1.00	5.55	294957
Spiegler et al.	Promodes-E	P	1.00	5.29	329206
Spiegler et al.	Promodes-H	P	1.00	6.01	239557
-	Morfessor Baseline	U	1.00	2.20	187194
-	Morfessor Categories-MAP	U	1.00	2.92	284560
-	letters	-	1.00	13.94	67

sor U+W, but none the differences between them was significant. With MBR combination, Morfessor Categories-MAP outperformed Baseline, but not with a statistical significance. Also the results of Iterative Compounding and Aggressive Compounding by Lignos, as well as grammatical morphemes, had no statistically significant difference to the results of Categories-MAP. Moreover, none of the submitted methods could improve the word-based baseline significantly.

4.4 Discussion

As the winners of the competitions varied, sometimes even within the same language, we studied the correlations of the different scores. Table 19 shows average correlations of the scores over the languages. In addition to the two linguistic evaluations—Morpho Challenge (MC) evaluation and EMMA—information retrieval evalation, and the two machine translation evaluations (single and combined systems), we included precision and recall of the linguistic evaluations and three statistics shown also in Tables 2–5. Although the number of algorithms (7–17) and the number of languages (1–4) is quite small, the strongest positive or negative correlations with a small deviation across the languages can provide us some insight on this variance.

Considering the linguistic evaluations, MC evaluation has a strong negative correlation (-0.67) between precision and recall. Recall has stronger correlation to the F-measure, likely due to that the variance in recall is usually larger than the variance in precision. In EMMA, precision and recall seem to be almost uncorrelated, and F-measure correlates better with precision than recall. Another interesting phenomenon is that the average number of morphs per word has a strong positive correlation ($+0.59$) to the recall value of the MC evaluation, and even stronger negative correlation (-0.76) to the precision value. In EMMA, precision has a negative correlation, although not as large (-0.46), but recall is almost uncorrelated. Thus, it seems that while F-measure of the MC evaluation can be improved by extensive segmentation, the same does not help with EMMA.

The amount of alternative analyses has a positive correlation ($+0.40$) to MC evaluation, but a negative one (-0.67) to EMMA. This corresponds to what has been observed before (Kurimo et al., 2009; Kurimo et al., 2010a; Spiegler and Monson, 2010): The recall in MC evaluation can be easily improved by including alternative analyses. This year, the DEAP algorithms had this advantage.

Comparing the information retrieval results of

Table 4: Statistics of the analyses for **German** words. Type indicates the level of supervision used in the method, #a/w is the average amount of analyses per word, #m/w is the average amount of morphemes per analysis, and #lexicon is the size of the morph lexicon.

Author	Method	Type	#a/w	#m/w	#lexicon
Kohonen et al.	Morfessor U+W	P	1.00	3.37	10902
Lignos	Aggressive Compounding	U	1.00	3.21	436916
Lignos	Base Inference	U	1.00	2.99	544704
Lignos	Iterative Compounding	U	1.00	3.05	549902
Nicolas et al.	MorphAcq	U	1.00	1.32	1888651
-	Morfessor Baseline	U	1.00	2.30	142352
-	Morfessor Categories-MAP	U	1.00	3.11	297425
-	letters	-	1.00	14.03	59

Competition 2 to Competition 1 results, it seems that good F-measure in Competition 1 linguistic evaluation does not necessarily mean good MAP in Competition 2 IR task. Algorithms with relatively high recall but relatively low precision score high in terms of F-measure but do not seem to perform well for IR. Good precision in the linguistic evaluation is more important for IR than good recall. Further, the methods that have large morph lexicon have a strong negative correlation to MAP (-0.65). The best predictor of IR performance is the F-measure using the EMMA evaluation method with correlation of $+0.66$.

The two scores in the machine translation evaluation of Competition 3 have naturally a high correlation. The BLEU score without system combination has a strongest correlation, $+0.84$, to the precision in the MC evaluation. Also the correlation to the precision in EMMA is high ($+0.59$). With system combination, the correlations decrease somewhat, but the strongest, 0.64, is still to the precision in MC evalution. However, the correlation to the F-measure in MC evalation is negative (-0.30), whereas the correlation to the F-measure in EMMA is positive ($+0.48$). Thus, also the machine translation evaluation indicates that the EMMA is more relevant evaluation with respect to the applications than the original Morpho Challenge evaluation.

In the information retrieval task of Competition 2, several algorithms could beat the word based "dummy" baseline and some algorithms were very close to the language specific word normalizers and stemmers. This indicates that semi-supervised and unsupervised methods are usable approaches for IR.

It is hard to achieve statistically significant differences in the IR task with only 50–60 queries for each language. However, as the Competition 1 metrics do not always completely reflect the performance of the algorithms, it is important to test them in a real life application as well. The IR task is designed to treat all participant algorithms equally. One problematic issue is the automatic stop list generation that is needed to achieve decent IR performance when there is high frequency terms in the corpus. Using one fixed threshold for all submissions is quite simplistic, but in practice seems to work well. To keep the results comparable to previous years, the stop list method has not yet been revised. Removing morphs that are labeled as affixes did not offer any improvements (with or without stop list) for any method with such labels. Further, the results are robust with respect to the threshold parameter.

In Competition 3, the differences between the final results after the MBR-based system combination were very small. In the Finnish-to-English task, seven algorithms could improve the word-based baseline with a statistical significance. In the German-to-English task, only two algorithms—the Morfessor methods provided for reference—could do the same. Moreover, they were the two best also in the Finnish task. Especially Morfessor Baseline seems to be surprisingly hard to beat in machine translation, as it provided the best results also in Morpho Challenge 2009.

5 Conclusions

The Morpho Challenge 2010 has successfully collected and evaluated the latest developments in unsupervised and semi-supervised learning methods

Table 5: Statistics of the analyses for **Turkish** words. Type indicates the level of supervision used in the method, #a/w is the average amount of analyses per word, #m/w is the average amount of morphemes per analysis, and #lexicon is the size of the morph lexicon.

Author	Method	Type	#a/w	#m/w	#lexicon
Golénia et al.	MAGIP	S	1.00	4.99	99549
Kohonen et al.	Morfessor S+W	S	1.00	3.58	9398
Kohonen et al.	Morfessor S+W+L	S	1.00	3.56	14713
Kohonen et al.	Morfessor U+W	P	1.00	4.57	595
Lignos	Aggressive Compounding	U	1.00	3.08	212291
Lignos	Base Inference	U	1.00	2.00	285196
Lignos	Iterative Compounding	U	1.00	2.42	248172
Nicolas et al.	MorphAcq	U	1.00	2.31	180709
Spiegler et al.	DEAP MDL-CAT	S	4.88	2.59	891260
Spiegler et al.	DEAP MDL-NOCAT	S	3.17	2.67	631500
Spiegler et al.	DEAP PROB-CAT	S	4.68	2.30	1169357
Spiegler et al.	DEAP PROB-NOCAT	S	3.49	2.34	1023903
Spiegler et al.	Promodes	P	1.00	4.50	42277
Spiegler et al.	Promodes-E	P	1.00	4.97	20512
Spiegler et al.	Promodes-H	P	1.00	4.25	58742
-	Morfessor Baseline	U	1.00	2.14	53473
-	Morfessor Categories-MAP	U	1.00	2.64	114834
-	letters	-	1.00	9.99	33

for morpheme analysis. The participants of the challenge included 5 research teams that submitted a total of 15 algorithms for the evaluations. As in Morpho Challenge 2005–2009, the morpheme analyses were tested in several languages (Finnish, Turkish, English and German) and tasks (comparisons to grammatical morphemes and information retrieval and machine translation experiments). In addition to the large amount of unlabeled data in each language available for unsupervised learning, this year new labeled data was made available to enable semi-supervised learning. Two new ways were provided to analyse the results in addition to the main evaluation metrics used already in the previous Morpho Challenges (F-measure in linguistic evaluations, mean average precision in information retrievalt tasks, and BLEU in machine translation tasks): EMMA (Spiegler and Monson, 2010) is an alternative measure to make comparisons to grammatical morphemes. The correlation table (see Table 19) reveals which measures actually correlate best to the application specific measures and, more generally, which measures correlate with each other.

Acknowledgments

We thank all participants of the Morpho Challenge 2010 for their submissions and enthusiasm. We are grateful to Stefan Bordag, University of Leipzig, and CLEF for the data resources, and Ebru Arisoy for making the Turkish gold standard available to the Challenge. Graeme W. Blackwood, William Byrne, Oskar Kohonen and Laura Leppänen have given valuable help with the Challenge evaluations. This work was supported by the Academy of Finland in the project *Adaptive Informatics,* Graduate School of Language Technology in Finland, and in part by the IST Programme of the European Community, under the FP7 project EMIME (213845) and the Challenge Program of PASCAL2 Network of Excellence. This publication only reflects the authors' views. We acknowledge that access rights to data and other materials are restricted due to other commitments.

References

Mathias Creutz and Krista Lagus. 2002. Unsupervised discovery of morphemes. In *Proceedings of the Workshop on Morphological and Phonological Learning of ACL-02,* pages 21–30.

Table 6: The submitted morpheme analyses compared to the gold standard in **English** (Competition 1).

Author	Method	Type	Precision	Recall	F-measure
Kohonen et al.	Morfessor S+W	S	0.6562	0.6928	0.6740
Kohonen et al.	Morfessor S+W+L	S	0.6787	0.6643	0.6714
Lignos	Base Inference	U	0.8077	0.5376	0.6455
Lignos	Iterative Compounding	U	0.8027	0.5276	0.6367
Spiegler et al.	DEAP MDL-NOCAT	S	0.5144	0.8095	0.6291
Spiegler et al.	DEAP PROB-NOCAT	S	0.5852	0.6306	0.6070
Lignos	Aggressive Compounding	U	0.7145	0.5231	0.6040
Kohonen et al.	Morfessor U+W	P	0.6033	0.5955	0.5994
Spiegler et al.	DEAP MDL-CAT	S	0.4983	0.7508	0.5990
Nicolas et al.	MorphAcq	U	0.6783	0.5343	0.5978
Spiegler et al.	DEAP PROB-CAT	S	0.5500	0.5661	0.5579
-	Morfessor Baseline	U	0.8139	0.4170	0.5514
Spiegler et al.	Promodes	P	0.3959	0.6472	0.4913
Spiegler et al.	Promodes-E	P	0.4969	0.4622	0.4790
-	Morfessor Categories-MAP	U	0.8684	0.3003	0.4463
Golénia et al.	MAGIP	S	0.2988	0.7065	0.4200
Spiegler et al.	Promodes-H	P	0.2684	0.6369	0.3777
-	letters	-	0.0441	0.9988	0.0844

Mathias Creutz and Krista Lagus. 2005a. Inducing the morphological lexicon of a natural language from unannotated text. In *Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR'05)*, pages 106–113.

Mathias Creutz and Krista Lagus. 2005b. Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor. Technical Report A81, Publications in Computer and Information Science, Helsinki University of Technology. URL: http://www.cis.hut.fi/projects/morpho/.

Adrià de Gispert, Sami Virpioja, Mikko Kurimo, and William Byrne. 2009. Minimum bayes risk combination of translation hypotheses from alternative morphological decompositions. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 73–76, Boulder, USA, June. Association for Computational Linguistics.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the 10th Machine Translation Summit*, pages 79–86, Phuket, Thailand.

Shankar Kumar and William Byrne. 2004. Minimum Bayes-Risk decoding for statistical machine translation. In *Proceedings of Human Language Technologies: The 2004 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 169–176.

Mikko Kurimo, Mathias Creutz, and Matti Varjokallio. 2008. Morpho Challenge evaluation using a linguistic Gold Standard. In *Advances in Multilingual and MultiModal Information Retrieval, 8th Workshop of the Cross-Language Evaluation Forum, CLEF 2007, Budapest, Hungary, September 19-21, 2007, Revised Selected Papers*, Lecture Notes in Computer Science , Vol. 5152, pages 864–873. Springer.

Mikko Kurimo, Ville Turunen, and Matti Varjokallio. 2009. Overview of Morpho Challenge 2008. In *Evaluating systems for Multilingual and MultiModal Information Access, 9th Workshop of the Cross-Language Evaluation Forum, CLEF 2008, Aarhus, Denmark, September 17-19, 2008, Revised Selected Papers*, Lecture Notes in Computer Science , Vol. 5152. Springer.

Mikko Kurimo, Sami Virpioja, Ville Turunen, and Krista Lagus. 2010a. Morpho Challenge 2005–2010: Evaluations and results. In *Proceedings of the 11th Meeting of the ACL Special Interest Group on Computational Morphology and Phonology*, pages 87–95, Uppsala, Sweden, July. Association for Computational Linguistics.

Mikko Kurimo, Sami Virpioja, Ville T. Turunen, Graeme W. Blackwood, and William Byrne. 2010b. Overview of Morpho Challenge 2009. In *In Multilingual Information Access Evaluation Vol. I-II, 10th Workshop of the Cross-Language Evaluation Forum, CLEF 2009, Corfu, Greece, September 30 - October 2, 2009, Revised Selected Papers*, Lecture Notes in Computer Science , Vol. 5152. Springer.

Christian Monson, Jaime Carbonell, Alon Lavie, and Lori Levin. 2009. ParaMor and Morpho Challenge

Table 7: The submitted morpheme analyses compared to the gold standard in **Finnish** (Competition 1).

Author	Method	Type	Precision	Recall	F-measure
Spiegler et al.	DEAP MDL-NOCAT	S	0.5603	0.7071	0.6252
Spiegler et al.	DEAP MDL-CAT	S	0.5743	0.6659	0.6167
Kohonen et al.	Morfessor S+W+L	S	0.5838	0.6335	0.6076
Kohonen et al.	Morfessor S+W	S	0.5759	0.5521	0.5638
Spiegler et al.	DEAP PROB-NOCAT	S	0.5666	0.4479	0.5003
Kohonen et al.	Morfessor U+W	P	0.5697	0.4298	0.4900
Lignos	Aggressive Compounding	U	0.6307	0.3998	0.4894
Spiegler et al.	DEAP PROB-CAT	S	0.5193	0.4241	0.4669
Spiegler et al.	Promodes-E	P	0.4326	0.4573	0.4446
Spiegler et al.	Promodes	P	0.4159	0.4741	0.4431
Golénia et al.	MAGIP	S	0.3625	0.5373	0.4329
-	Morfessor Categories-MAP	U	0.8031	0.2951	0.4316
Spiegler et al.	Promodes-H	P	0.3508	0.5345	0.4236
Lignos	Iterative Compounding	U	0.7532	0.2670	0.3943
Lignos	Base Inference	U	0.7898	0.2446	0.3735
-	Morfessor Baseline	U	0.9060	0.1439	0.2483
-	letters	-	0.0504	0.9989	0.0960

Table 8: The submitted morpheme analyses compared to the gold standard in **German** (Competition 1).

Author	Method	Type	Precision	Recall	F-measure
Kohonen et al.	Morfessor U+W	P	0.5855	0.4494	0.5085
-	Morfessor Categories-MAP	U	0.7270	0.3543	0.4764
Lignos	Base Inference	U	0.6638	0.3536	0.4614
Lignos	Aggressive Compounding	U	0.5941	0.3721	0.4576
Lignos	Iterative Compounding	U	0.6213	0.3470	0.4453
Nicolas et al.	MorphAcq	U	0.6997	0.2529	0.3715
-	Morfessor Baseline	U	0.8280	0.1977	0.3192
-	letters	-	0.0280	0.9992	0.0545

2008. In *Evaluating systems for Multilingual and MultiModal Information Access, 9th Workshop of the Cross-Language Evaluation Forum, CLEF 2008, Aarhus, Denmark, September 17-19, 2008, Revised Selected Papers*, Lecture Notes in Computer Science , Vol. 5152. Springer.

P. Ogilvie and J. Callan. 2002. Experiments using the lemur toolkit. In *Proc. TREC '01*, pages 103–108, Gaithersburg, Maryland, USA. National Institute of Standards and Technology, special publication.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL'02)*, pages 311–318, Morristown, NJ, USA. Association for Computational Linguistics.

Sebastian Spiegler and Christian Monson. 2010. EMMA: A novel evaluation metric for morphological analysis. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*, August.

Table 9: The submitted morpheme analyses compared to the gold standard in **Turkish** (Competition 1).

Author	Method	Type	Precision	Recall	F-measure
Kohonen et al.	Morfessor S+W+L	S	0.7169	0.5997	0.6531
Spiegler et al.	DEAP MDL-NOCAT	S	0.6893	0.5612	0.6187
Spiegler et al.	DEAP MDL-CAT	S	0.7275	0.5349	0.6165
Spiegler et al.	DEAP PROB-CAT	S	0.6886	0.5581	0.6165
Spiegler et al.	DEAP PROB-NOCAT	S	0.6571	0.5804	0.6163
Kohonen et al.	Morfessor S+W	S	0.6571	0.4715	0.5490
Spiegler et al.	Promodes	P	0.4659	0.5167	0.4900
Spiegler et al.	Promodes-E	P	0.4075	0.5239	0.4584
-	Morfessor Categories-MAP	U	0.7938	0.3188	0.4549
Kohonen et al.	Morfessor U+W	P	0.4071	0.4676	0.4352
Spiegler et al.	Promodes-H	P	0.4788	0.3937	0.4321
Golénia et al.	MAGIP	S	0.3200	0.6580	0.4306
Lignos	Aggressive Compounding	U	0.5551	0.3436	0.4245
Lignos	Iterative Compounding	U	0.6869	0.2144	0.3268
Nicolas et al.	MorphAcq	U	0.7902	0.1978	0.3164
-	Morfessor Baseline	U	0.8968	0.1778	0.2967
Lignos	Base Inference	U	0.7281	0.1611	0.2638
-	letters	-	0.0866	0.9913	0.1593

Table 10: The submitted morpheme analyses compared to the gold standard in **English** using EMMA.

Author	Method	Type	Precision	Recall	F-measure
Kohonen et al.	Morfessor S+W+L	S	0.8208	0.8450	0.8327
Lignos	Base Inference	U	0.8744	0.7682	0.8178
Lignos	Iterative Compounding	U	0.8736	0.7641	0.8152
Kohonen et al.	Morfessor S+W	S	0.7939	0.8176	0.8056
Lignos	Aggressive Compounding	U	0.8217	0.7588	0.7890
-	Morfessor Baseline	U	0.8686	0.7226	0.7889
-	Morfessor Categories-MAP	U	0.9209	0.6815	0.7833
Nicolas et al.	MorphAcq	U	0.7851	0.7759	0.7804
Kohonen et al.	Morfessor U+W	P	0.7313	0.7673	0.7489
Spiegler et al.	Promodes-E	P	0.6573	0.6977	0.6769
Spiegler et al.	Promodes	P	0.5658	0.7382	0.6406
Spiegler et al.	DEAP MDL-NOCAT	S	0.5233	0.8159	0.6376
Spiegler et al.	DEAP PROB-NOCAT	S	0.5618	0.7314	0.6355
Golénia et al.	MAGIP	S	0.4876	0.7083	0.5775
Spiegler et al.	Promodes-H	P	0.4016	0.6298	0.4904
Spiegler et al.	DEAP MDL-CAT	S	0.2579	0.8503	0.3957
Spiegler et al.	DEAP PROB-CAT	S	0.2229	0.7730	0.3460
-	letters	-	0.0942	0.2953	0.1429

Table 11: The submitted morpheme analyses compared to the gold standard in **Finnish** using EMMA.

Author	Method	Type	Precision	Recall	F-measure
Kohonen et al.	Morfessor S+W+L	S	0.6717	0.7573	0.7119
Lignos	Base Inference	U	0.7990	0.5605	0.6588
Lignos	Iterative Compounding	U	0.7733	0.5692	0.6557
Lignos	Aggressive Compounding	U	0.6691	0.6004	0.6329
Kohonen et al.	Morfessor S+W	S	0.5914	0.6533	0.6208
-	Morfessor Categories-MAP	U	0.7005	0.5425	0.6114
-	Morfessor Baseline	U	0.7614	0.4742	0.5844
Kohonen et al.	Morfessor U+W	P	0.5358	0.5571	0.5462
Spiegler et al.	Promodes	P	0.4493	0.5947	0.5118
Spiegler et al.	Promodes-E	P	0.4423	0.5555	0.4925
Golénia et al.	MAGIP	S	0.4349	0.5364	0.4803
Spiegler et al.	DEAP PROB-NOCAT	S	0.3936	0.5622	0.4629
Spiegler et al.	Promodes-H	P	0.3589	0.5273	0.4271
Spiegler et al.	DEAP MDL-NOCAT	S	0.2960	0.6645	0.4095
Spiegler et al.	DEAP MDL-CAT	S	0.2101	0.6426	0.3166
Spiegler et al.	DEAP PROB-CAT	S	0.2106	0.5095	0.2979
-	letters	-	0.0929	0.3046	0.1424

Table 12: The submitted morpheme analyses compared to the gold standard in **German** using EMMA.

Author	Method	Type	Precision	Recall	F-measure
-	Morfessor Categories-MAP	U	0.7826	0.5583	0.6517
Nicolas et al.	MorphAcq	U	0.7611	0.5126	0.6125
Lignos	Base Inference	U	0.7498	0.5084	0.6059
-	Morfessor Baseline	U	0.8128	0.4806	0.6040
Kohonen et al.	Morfessor U+W	P	0.6571	0.5560	0.6023
Lignos	Iterative Compounding	U	0.7155	0.5035	0.5910
Lignos	Aggressive Compounding	U	0.6875	0.5117	0.5867
-	letters	-	0.0834	0.2397	0.1237

Table 13: The submitted morpheme analyses compared to the gold standard in **Turkish** using EMMA.

Author	Method	Type	Precision	Recall	F-measure
Kohonen et al.	Morfessor S+W+L	S	0.7532	0.5779	0.6539
Kohonen et al.	Morfessor S+W	S	0.5734	0.4425	0.4994
-	Morfessor Categories-MAP	U	0.6468	0.4004	0.4946
Lignos	Base Inference	U	0.7131	0.3530	0.4722
Nicolas et al.	MorphAcq	U	0.6310	0.3744	0.4699
Lignos	Iterative Compounding	U	0.6607	0.3602	0.4662
Lignos	Aggressive Compounding	U	0.5709	0.3733	0.4513
-	Morfessor Baseline	U	0.6543	0.3430	0.4500
Spiegler et al.	DEAP MDL-NOCAT	S	0.3590	0.5554	0.4360
Spiegler et al.	Promodes	P	0.4471	0.4117	0.4286
Spiegler et al.	DEAP PROB-NOCAT	S	0.3203	0.5223	0.3971
Spiegler et al.	Promodes-H	P	0.4123	0.3581	0.3833
Spiegler et al.	Promodes-E	P	0.3722	0.3664	0.3693
Golénia et al.	MAGIP	S	0.3479	0.3541	0.3509
Spiegler et al.	DEAP MDL-CAT	S	0.2390	0.5564	0.3344
Spiegler et al.	DEAP PROB-CAT	S	0.2417	0.5206	0.3301
Kohonen et al.	Morfessor U+W	P	0.2959	0.2627	0.2783
-	letters	-	0.1121	0.2092	0.1459

Table 14: The obtained mean average precision (MAP) in the information retrieval task for **English**. Submissions with no statistically significant difference to the top performer are marked with X.

Author	Method	Type	MAP
-	snowball porter	-	0.4092 X
-	Best Prev. (2008 Monson P+M)	U	0.4022 X
-	TWOL first	-	0.4020 X
-	TWOL all	-	0.3994 X
-	Morfessor Baseline	U	0.3835 X
Lignos	Base Inference	U	0.3832 X
Kohonen et al.	Morfessor S+W+L	S	0.3820 X
Lignos	Aggressive Compounding	U	0.3784 X
-	Grammatical Morphemes (First)	-	0.3777 X
Kohonen et al.	Morfessor S+W	S	0.3776 X
Lignos	Iterative Compounding	U	0.3776 X
-	Morfessor Categories-MAP	U	0.3754 X
Kohonen et al.	Morfessor U+W	P	0.3690 X
Nicolas et al.	MorphAcq	U	0.3680 X
-	Grammatical Morphemes (All)	-	0.3567 X
Spiegler et al.	Promodes-E	P	0.3379
-	dummy	-	0.3304
Spiegler et al.	Promodes	P	0.3286
Spiegler et al.	DEAP PROB-NOCAT	S	0.3141
Spiegler et al.	DEAP MDL-NOCAT	S	0.2916
Spiegler et al.	DEAP MDL-CAT	S	0.2409
Golénia et al.	MAGIP	S	0.2257
Spiegler et al.	DEAP PROB-CAT	S	0.2186
Spiegler et al.	Promodes-H	P	0.2034

Table 15: The obtained mean average precision (MAP) in the information retrieval task for **Finnish**. Submissions with no statistically significant difference to the top performer are marked with X.

Author	Method	Type	MAP
-	TWOL first	-	0.4973 X
Lignos	Aggressive Compounding	U	0.4914 X
-	Best Prev. (2008 McNamee four)	-	0.4911 X
-	TWOL all	-	0.4801 X
-	Morfessor Categories-MAP	U	0.4754 X
Kohonen et al.	Morfessor S+W	S	0.4750 X
Kohonen et al.	Morfessor S+W+L	S	0.4465 X
-	Grammatical Morphemes (First)	-	0.4317 X
-	snowball finnish	-	0.4275 X
-	Morfessor Baseline	U	0.4235 X
Lignos	Iterative Compounding	U	0.4229 X
Spiegler et al.	DEAP MDL-NOCAT	S	0.4160 X
Lignos	Base Inference	U	0.4151 X
-	Grammatical Morphemes (All)	-	0.4093 X
Kohonen et al.	Morfessor U+W	P	0.4042 X
Golénia et al.	MAGIP	S	0.3871 X
Spiegler et al.	DEAP MDL-CAT	S	0.3725 X
Spiegler et al.	Promodes	P	0.3721 X
-	dummy	-	0.3529
Spiegler et al.	Promodes-E	P	0.3213
Spiegler et al.	DEAP PROB-NOCAT	S	0.3174
Spiegler et al.	Promodes-H	P	0.3109
Spiegler et al.	DEAP PROB-CAT	S	0.2843

Table 16: The obtained mean average precision (MAP) in the information retrieval task for **German**. Submissions with no statistically significant difference to the top performer are marked with X.

Author	Method	Type	MAP
-	TWOL first	-	0.4836 X
-	TWOL all	-	0.4745 X
-	Best Prev. (2008 Monson P+M)	U	0.4734 X
Kohonen et al.	Morfessor U+W	P	0.4666 X
-	Morfessor Categories-MAP	U	0.4657 X
Lignos	Base Inference	U	0.4434 X
-	Morfessor Baseline	U	0.4391 X
Lignos	Aggressive Compounding	U	0.4379 X
Lignos	Iterative Compounding	U	0.4200 X
Nicolas et al.	MorphAcq	U	0.3862
-	snowball german	-	0.3859
-	dummy	-	0.3508
-	Grammatical Morphemes (First)	-	0.3350
-	Grammatical Morphemes (All)	-	0.3008

Table 17: The results of the submitted morpheme analyses used in machine translation from **Finnish**. Results with no statistically significant difference to the top performer are marked with X. Results with no statistically significant difference to those of the word-based model are marked with Y.

Author	Method	Type	BLEU
MBR combination with word-based model			
-	Morfessor Baseline	U	0.2665 X
-	Morfessor Categories-MAP	U	0.2634 X
-	Grammatical Morphemes	-	0.2623
Lignos	Base Inference	U	0.2619
Lignos	Iterative Compounding	U	0.2605
Lignos	Aggressive Compounding	U	0.2597 Y
Kohonen et al.	Morfessor S+W	S	0.2594
Spiegler et al.	DEAP MDL-CAT	S	0.2590
Spiegler et al.	DEAP PROB-CAT	S	0.2589 Y
Kohonen et al.	Morfessor S+W+L	S	0.2582 Y
Spiegler et al.	Promodes-E	P	0.2579 Y
Spiegler et al.	Promodes	P	0.2564 Y
Spiegler et al.	DEAP MDL-NOCAT	S	0.2562 Y
Kohonen et al.	Morfessor U+W	P	0.2561 Y
Spiegler et al.	DEAP PROB-NOCAT	S	0.2555 Y
Spiegler et al.	Promodes-H	P	0.2535 Y
Golénia et al.	MAGIP	S	0.2533 Y
Individual systems			
-	Morfessor Baseline	U	0.2614 X
-	words	-	0.2543 X Y
Lignos	Base Inference	U	0.2504 Y
-	Morfessor Categories-MAP	U	0.2493 Y
-	Grammatical Morphemes	-	0.2492 Y
Lignos	Iterative Compounding	U	0.2463 Y
Kohonen et al.	Morfessor U+W	P	0.2456 Y
Lignos	Aggressive Compounding	U	0.2444
Spiegler et al.	DEAP MDL-NOCAT	S	0.2404
Spiegler et al.	DEAP PROB-NOCAT	S	0.2383
Kohonen et al.	Morfessor S+W	S	0.2376
Kohonen et al.	Morfessor S+W+L	S	0.2339
Spiegler et al.	DEAP PROB-CAT	S	0.2316
Spiegler et al.	DEAP MDL-CAT	S	0.2303
Spiegler et al.	Promodes-E	P	0.2251
Spiegler et al.	Promodes	P	0.2192
Spiegler et al.	Promodes-H	P	0.2107
Golénia et al.	MAGIP	S	0.2027

Table 18: The results of the submitted morpheme analyses used in machine translation from **German**. Results with no statistically significant difference to the top performer are marked with X. Results with no statistically significant difference to those of the word-based model are marked with Y. In addition, a group of methods with no statistically significant differences between any of their results are marked with Z.

Author	Method	Type	BLEU
MBR combination with word-based model			
-	Morfessor Categories-MAP	U	0.3008 X
-	Morfessor Baseline	U	0.2994 X
Kohonen et al.	Morfessor U+W	P	0.2986 Y
Nicolas et al.	MorphAcq	U	0.2986 Y
Lignos	Iterative Compounding	U	0.2985 X Y
Lignos	Aggressive Compounding	U	0.2980 X Y
-	Grammatical Morphemes	-	0.2970 X Y
Lignos	Base Inference	U	0.2952 Y
Individual systems			
-	Morfessor Baseline	U	0.2963 X Y
-	words	-	0.2955 X Y
-	Morfessor Categories-MAP	U	0.2862
Kohonen et al.	Morfessor U+W	P	0.2817 Z
Lignos	Base Inference	U	0.2803 Z
Lignos	Aggressive Compounding	U	0.2788 Z
-	Grammatical Morphemes	-	0.2782 Z
Lignos	Iterative Compounding	U	0.2770 Z
Nicolas et al.	MorphAcq	U	0.2751 Z

Table 19: Correlations between evaluation measures and some statistics of the methods. The included methods are all those submitted to the competition and the reference methods Morfessor Baseline and Morfessor Categories-MAP. Correlations were calculated separately for each language, and the average over the languages is shown. The number in parentheses is the corresponding standard deviation, and the subscript shows the number of languages that could be applied. MC-P, MC-R, and MC-F are the precision, recall and F-measure of the standard linguistic evaluation. EMMA-P, EMMA-R, and EMMA-F are the precision, recall and F-measure of the EMMA evaluation. IR-MAP is the mean average precision in the information retrieval task. S-BLEU is the BLEU score of the single system in machine translation task, and C-BLEU is the BLEU score of the combined system. The last three are statistics: size of the morpheme lexicon (#lexicon), average number of morphemes per word (#m/w), and average number of analyses per word (#a/w).

	MC-P	MC-R	MC-F	EMMA-P	EMMA-R	EMMA-F
MC-P	$+1.00\,(0.00)_4$	$-0.67\,(0.14)_4$	$-0.17\,(0.56)_4$	$+0.73\,(0.19)_4$	$+0.02\,(0.34)_4$	$+0.54\,(0.11)_4$
MC-R	$-0.67\,(0.14)_4$	$+1.00\,(0.00)_4$	$+0.74\,(0.33)_4$	$-0.69\,(0.11)_4$	$+0.61\,(0.11)_4$	$-0.29\,(0.17)_4$
MC-F	$-0.17\,(0.56)_4$	$+0.74\,(0.33)_4$	$+1.00\,(0.00)_4$	$-0.35\,(0.45)_4$	$+0.82\,(0.04)_4$	$+0.06\,(0.27)_4$
EMMA-P	$+0.73\,(0.19)_4$	$-0.69\,(0.11)_4$	$-0.35\,(0.45)_4$	$+1.00\,(0.00)_4$	$-0.16\,(0.18)_4$	$+0.82\,(0.21)_4$
EMMA-R	$+0.02\,(0.34)_4$	$+0.61\,(0.11)_4$	$+0.82\,(0.04)_4$	$-0.16\,(0.18)_4$	$+1.00\,(0.00)_4$	$+0.29\,(0.20)_4$
EMMA-F	$+0.54\,(0.11)_4$	$-0.29\,(0.17)_4$	$+0.06\,(0.27)_4$	$+0.82\,(0.21)_4$	$+0.29\,(0.20)_4$	$+1.00\,(0.00)_4$
IR-MAP	$+0.42\,(0.46)_3$	$-0.03\,(0.54)_3$	$+0.40\,(0.27)_3$	$+0.48\,(0.59)_3$	$+0.39\,(0.17)_3$	$+0.66\,(0.32)_3$
S-BLEU	$+0.84\,(0.11)_2$	$-0.51\,(0.10)_2$	$-0.33\,(0.15)_2$	$+0.59\,(0.01)_2$	$-0.12\,(0.10)_2$	$+0.37\,(0.13)_2$
C-BLEU	$+0.64\,(0.38)_2$	$-0.44\,(0.36)_2$	$-0.30\,(0.20)_2$	$+0.45\,(0.25)_2$	$+0.07\,(0.37)_2$	$+0.48\,(0.07)_2$
#lexicon	$+0.04\,(0.17)_4$	$+0.02\,(0.30)_4$	$+0.11\,(0.37)_4$	$-0.39\,(0.39)_4$	$+0.13\,(0.40)_4$	$-0.42\,(0.33)_4$
#m/w	$-0.76\,(0.17)_4$	$+0.59\,(0.14)_4$	$+0.13\,(0.54)_4$	$-0.46\,(0.25)_4$	$+0.02\,(0.37)_4$	$-0.27\,(0.31)_4$
#a/w	$-0.02\,(0.22)_3$	$+0.37\,(0.08)_3$	$+0.40\,(0.27)_3$	$-0.72\,(0.05)_3$	$+0.38\,(0.32)_3$	$-0.67\,(0.23)_3$

	IR-MAP	S-BLEU	C-BLEU	#lexicon	#m/w	#a/w
MC-P	$+0.42\,(0.46)_3$	$+0.84\,(0.11)_2$	$+0.64\,(0.38)_2$	$+0.04\,(0.17)_4$	$-0.76\,(0.17)_4$	$-0.02\,(0.22)_3$
MC-R	$-0.03\,(0.54)_3$	$-0.51\,(0.10)_2$	$-0.44\,(0.36)_2$	$+0.02\,(0.30)_4$	$+0.59\,(0.14)_4$	$+0.37\,(0.08)_3$
MC-F	$+0.40\,(0.27)_3$	$-0.33\,(0.15)_2$	$-0.30\,(0.20)_2$	$+0.11\,(0.37)_4$	$+0.13\,(0.54)_4$	$+0.40\,(0.27)_3$
EMMA-P	$+0.48\,(0.59)_3$	$+0.59\,(0.01)_2$	$+0.45\,(0.25)_2$	$-0.39\,(0.39)_4$	$-0.46\,(0.25)_4$	$-0.72\,(0.05)_3$
EMMA-R	$+0.39\,(0.17)_3$	$-0.12\,(0.10)_2$	$+0.07\,(0.37)_2$	$+0.13\,(0.40)_4$	$+0.02\,(0.37)_4$	$+0.38\,(0.32)_3$
EMMA-F	$+0.66\,(0.32)_3$	$+0.37\,(0.13)_2$	$+0.48\,(0.07)_2$	$-0.42\,(0.33)_4$	$-0.27\,(0.31)_4$	$-0.67\,(0.23)_3$
IR-MAP	$+1.00\,(0.00)_3$	$+0.49\,(0.03)_2$	$+0.32\,(0.23)_2$	$-0.65\,(0.22)_3$	$-0.04\,(0.80)_3$	$-0.52\,(0.10)_2$
S-BLEU	$+0.49\,(0.03)_2$	$+1.00\,(0.00)_2$	$+0.60\,(0.29)_2$	$-0.33\,(0.36)_2$	$-0.33\,(0.53)_2$	$-0.07\,(0.00)_1$
C-BLEU	$+0.32\,(0.23)_2$	$+0.60\,(0.29)_2$	$+1.00\,(0.00)_2$	$-0.11\,(0.02)_2$	$-0.35\,(0.36)_2$	$-0.07\,(0.00)_1$
#lexicon	$-0.65\,(0.22)_3$	$-0.33\,(0.36)_2$	$-0.11\,(0.02)_2$	$+1.00\,(0.00)_4$	$-0.47\,(0.33)_4$	$+0.89\,(0.05)_3$
#m/w	$-0.04\,(0.80)_3$	$-0.33\,(0.53)_2$	$-0.35\,(0.36)_2$	$-0.47\,(0.33)_4$	$+1.00\,(0.00)_4$	$-0.18\,(0.37)_3$
#a/w	$-0.52\,(0.10)_2$	$-0.07\,(0.00)_1$	$-0.07\,(0.00)_1$	$+0.89\,(0.05)_3$	$-0.18\,(0.37)_3$	$+1.00\,(0.00)_3$

Morphological analysis using morpheme graph and mixed-integer computation of stable models

Bruno Golénia, Sebastian Spiegler, Oliver Ray and Peter Flach
Intelligent Systems Group, University of Bristol, UK
{goleniab, spiegler, oray, flach}@cs.bris.ac.uk

Abstract

This paper presents a language-independent supervised word decomposition algorithm called MAGIP (Morphological Analysis by morpheme Graph and mixed-Integer Programming). MAGIP proceeds in two steps. First, a morpheme graph is created using a training set of 1,000 morpheme sequences and parses are generated for a second large set of words. Secondly, parses are selected through the computation of stable models by minimising inference of new morphemes. We evaluated MAGIP on three languages (English, Turkish and Finnish) using the Morpho Challenge evaluation scripts and obtained an F-measure around 0.37 for each language, with precision around 0.26 and recall around 0.69 for English and Turkish and 0.53 for Finnish.

1 Introduction

Morphological analysis is concerned with the process of segmenting and labeling a given corpus of words into a set of morphemes. Morphemes are the smallest units bearing a meaning in a word. In a corpus, the quantity of different morphemes is usually smaller than the quantity of different words. Labeling a morpheme provides information on the role of the morpheme in a word (eg. verb, noun, plural and so forth). In this paper, we are focusing on the segmentation of large corpus (English, Turkish and Finnish). To do so, we assume that a small dataset of words with their label sequences of morphemes is given. We divide this dataset into a training and a development dataset. Using the training dataset, we create a morpheme graph to generate potential parses. Then, we select a parse for each word by computing stable models using the development dataset. The computation of stable models is carried out by the reduction to integer programming. The remainder of the paper is structured as follows. Firstly, the generation of morphemes using a morpheme graph is presented. Secondly, the selection of parses by computing stable models is introduced. Finally, we show the results and we conclude.

2 Generate Parses Using a Morpheme Graph

In order to generate parses of a given word, we decided to use a morpheme graph to represent morphemes and their relationship (Golénia et al., 2010). We will refer to the training dataset of morpheme sequences in which only segmentation information is taken in consideration as M-corpus. The morpheme graph is defined as follows:

Definition 1 *Let* $M = \{m_i | 1 \leq i \leq z\}$ *be a set of morphemes, let* f_i *be the frequency (total number of times) with which morpheme* m_i *occurs in the M-corpus, let* $v_i = (m_i, f_i)$ *for* $1 \leq i \leq z$*, and let* $f_{i,j}$ *denote the frequency in the M-corpus in which morpheme* m_i *is followed by morpheme* m_j*. The* morpheme graph $G_m = (V, E)$ *is a directed graph with vertices* $V = \{v_i | 1 \leq i \leq z\}$ *and edges* $E = \{(v_i, v_j) | f_{i,j} > 0\}$*. We treat* $f_{i,j}$ *as the label of the edge from* v_i *to* v_j*.*

To select parses of a given word in a corpus, we use the morpheme graph with the formula $MGraph$ developed in (Shalonova and Golénia, 2010) for generating prefix sequences and suffix sequences during the test phase of the algorithm GBUSS. $MGraph$ is defined as follows:

Definition 2 *We define* $MGraph$ *of a morpheme sequence without boundaries* x *as follows:*

$$MGraph(x) = \arg\max_{t \subseteq x} \frac{1}{N_t - C_t} \sum_{m \in t} L_m log(f_m+1) \quad (1)$$

where

- *t is a morpheme sequence with boundaries of x,*
- *m is a morpheme of t,*
- *f_m is the frequency of the morpheme m,*
- *N_t is the number of morphemes taken in the graph,*
- *C_t is the number of morphemes taken and contiguous in the graph.*

Note that $MGraph$ does not use the frequency of the neighborhood. We leave it as future work. In the next section, we present an optimisation procedure to compute stable models in order to select a set of parses in a corpus.

3 Selecting Parses by Computing Stable Models

Currently, we have selected a set of parses for each unseen word using a morpheme graph derived from training set of morpheme sequences. Now, the problem is to select which parse is potentially good for each word of a corpus by inferring the existence of new morphemes. We are focusing on selecting the parses which introduce as few new morphemes as possible. Thus, for each parse selected by $MGraph$, we are taking into account only the morphemes not taken in the graph. Subsequently, we show how this problem can be seen as computing stable models of a normal logic program which can then be transformed into an integer program. We propose a generalisation of Bell et al.'s method to compute stable models adapted to the selection of parses from a corpus. This section is contains two parts. Firstly, we give the necessary logical background and define the stable model semantic. Secondly, we describe Bell et al.'s method and our generalisation of it.

3.1 Normal Logic Programming

In this part, we introduce the usual notation used in normal logic programming. We call *atom* $p(t_1, ..., t_n)$ a predicate p consisting of n terms ($n \geq 0$). A *term* is either a variable, a constant or a function of terms. We define a ground atom as an atom without variables. A literal l is an atom a (positive literal) or the negation of an atom $not\ a$ (negative literal). Here, not is the Negation As Failure (NAF) (K. Clark, 1977). A clause $h \leftarrow l_1 \wedge ... \wedge l_q$ is composed of two parts (head and body) where the *head* is a positive literal h and the *body* is a set of literals $l_1 \wedge ... \wedge l_q$. In our notation, $\wedge$ means the conjunction, $\leftarrow$ the implication and q the number of body literals. A *fact* is a clause without body ($q = 0$). An *integrity constraint* is a clause without head.

Definition 3 (Normal logic program) *A normal logic program P is a finite set of clauses of the following form*

$$h \leftarrow p_1 \wedge ... \wedge p_l \wedge not\ d_1 \wedge ... \wedge not\ d_k$$

where

- *$p_1 \wedge ... \wedge p_l$ and $d_1 \wedge ... \wedge d_k$ are atoms,*
- *$l \geq 0$ and $k \geq 0$.*

To interpret a normal logic program P, we use Herbrand interpretations. A Herbrand interpretation I is a set of ground atoms. The set of all ground atoms of P is called the Herbrand base $HB(P)$. The Herbrand universe $HU(P)$ is set of all ground terms of P. A model M is an interpretation that satisfies P. A model M is a minimal model if and only there is no model N such that $N \subset M$. The definition of stable model model is described as follows:

Definition 4 (Stable model) *A model M is stable if M is the minimal model for the Gelfond-Lifschitz reduction obtained by deleting from P (M. Gelfond and V. Lifschitz, 1988):*

- *each rule where at least a d_i is in M,*
- *all $not\ d_i$ from the remaining rules.*

3.2 Mixed-Integer Programming

In this part, we present Bell et al.'s method to compute stable models. Then, we generalise Bell's method in three ways. Bell et al. defined an optimisation problem to compute minimal models based on an integer program as follows (C. Bell, A. Nerode, R. Ng and V. Subrahmanian, 1994):

Definition 5 (Minimisation model optimisation problem) *Given a logic program P, a minimisation model optimisation problem $\Pi(P)$ is the optimisation of an integer program with the following objective function:*

$$min \sum_{a \in HB(P)} x_a$$

w.r.t the following constraint:

$$x_{h_i} - (\sum_{f=1}^{k_i} x_{p_{if}}) + (\sum_{f=1}^{l_i} x_{d_{if}}) \geq 1 - k_i$$

for each clause

$$h_i \leftarrow p_{i1} \wedge ... \wedge p_{ik_i} \wedge not\ d_{i1} \wedge ... \wedge not\ d_{il_i}$$

Now, we generalise Bell et al.'s formalism to "if and only if" clauses. Those rules are useful to express that a parse is possible if and only if all of its morphemes (not taken in the graph) are possible. A "if and only if" clause i is defined as follows:

$$h_i \iff p_{i1} \wedge ... \wedge p_{ik_i} \wedge not\ d_{i1} \wedge ... \wedge not\ d_{il_i}$$

A "if and only if" clause i can be formulated for an optimisation problem $\Pi(P)$ as follows:

$$q_i x_{h_i} - (\sum_{f=1}^{k_i} x_{p_{if}}) + (\sum_{f=1}^{l_i} x_{d_{if}}) \leq l_i$$

We now introduce a clause with "disjunctive body literals". Those rules permit to indicate that a word is possible if at least one parse is possible. A clause i with "disjunctive body literals" is represented as follows:

$$h_i \leftarrow \bigvee_{j=1}^{y_i} p_{j1} \wedge ... \wedge p_{jk_j} \wedge not\ d_{j1} \wedge ... \wedge not\ d_{jl_j}$$

Here, y_i defines the number of disjunctions in the body. This clause is represented in $\Pi(P)$ by the following inequality:

$$y_i x_{h_i} - (\sum_{j=1}^{y_i} \sum_{f=1}^{k_j} x_{p_{jf}}) + (\sum_{j=1}^{y_i} \sum_{f=1}^{l_j} x_{d_{if}}) \geq y_i - \sum_{j=1}^{y_i} k_j$$

Now, in order to give priority on the stable models, we introduce the minimisation rules which allows us to give some preferences on the search of solutions. Thus, a minimisation rule permits to find stable models with as few of the given literals as possible. This is quite useful since the amount of solutions is intractable. Those rules change just the coefficients of the objective function in an integer program. A Minimisation rule i is represented as follows:

$$minimize\ \{\alpha_{i1} p_{i1}, ..., \alpha_{ik_i} p_{ik_i}, \beta_{i1} not\ d_{i1}, ..., \beta_{il_i} not\ d_{il_i}\}$$

Here $\alpha_{i1}, ..., \alpha_{ik_i}$ and β_{il_i} ... β_{il_i} are weights value > 0 which can be real or integer. To keep consistency of the meaning of the minimisation rule in the integer programming statement, we need to avoid weights between $]0,1[$. For instance for α_i, we take the lowest value $\alpha_{i1}, ..., \alpha_{ik_i} > 0$ and multiply all $\alpha_{i1}, ..., \alpha_{ik_i}$ by the inverse of its value. We do the same for β_i if $\exists\ \beta_{il_i}, ..., \beta_{il_i} \in]0,1[$. Let $OPTSET = \{HB(P) \setminus \{\cup_{f=1}^{l_i} x_{d_{if}} \cup_{f=1}^{k_i} x_{p_{if}}\}\}$ the set difference between the variables in the objective function and the minimisation rule i. Let $B_i = \sum_{f=1}^{l_i} \beta_{if}$ the sum of coefficients in the negative part of the minimisation rule i. Let $\{c_a | a \in HB(P)\}$ the coefficients of the objective function, initialised at 1 and $C_i = \sum_{a \in OPTSET} c_a$ the sum of coefficients for the set difference. The new objective function with respect to the minimisation rule i is:

$$min \sum_{a \in OPTSET} (B_i + 1) c_a x_a + \sum_{f=1}^{k_i} (B_i + 1)(C_i + 1) \alpha_{if} c_{if} x_{p_{if}} + \sum_{f=1}^{l_i} \beta_{if} c_{if} x_{d_{if}}$$

Note that this transformation can be simplified if $l_i = 0$ or $k_i = 0$. Also, note that the coefficients α_i and β_i can be non-integer and therefore the problem may be a mixed-integer program. Similarly, the maximisation rule can be derived directly. Nevertheless, afterwards we will not serve of the maximisation rule. As it can be remarked the minimisation rules can be incorporated into a hierarchy which is quite powerful and even be mixed with maximisation rules. However, multiple minimisation/maximisation rules in a hierarchy can create an overflow due to too large value for the coefficients. Finally, to select parses for all unseen words, we defined a normal logic program. Here, we provide an example with 2 words having each of them at most 2 parses:

Example 3.1 $p_1 \iff m_1 \wedge m_2$
$p_2 \iff m_1 \wedge m_3$
$p_3 \iff m_2 \wedge m_3$
$p_4 \iff m_4$
$w_1 \leftarrow p_1 \vee p_2$
$w_2 \leftarrow p_3 \vee p_4$
$false \leftarrow not\ w_1$
$false \leftarrow not\ w_2$
$minimize\{1/(f(m_1)L_{m_1})m_1, 1/(f(m_1)L_{m_2})m_2, 1/(f(m_3)L_{m_3})m_3, 1/(f(m_4)L_{m_4})m_4\}$
Here p stands for parse, m for morpheme. Then, $f(m_i)$ and L_{m_i} stands for the frequency and length respectively of the morpheme i in the optimisation problem. The idea behind the minimisation rule is to have preferences on morphemes witch have a high length and high frequency.

We call MAGIP (Morphological Analysis by morpheme Graph and mixed-Integer Programming) the algorithm which combines the selection of parses using the morpheme graph presented in the previous section and the optimisation of a mixed-integer program presented in this section.

4 Related Work

Learning in natural language processing using logic programming techniques is not new. Inductive Logic Program (ILP) systems such as GOLEM (Muggleton and Feng, 1990), FOIL

(Quinlan and Mostow, 1990) and then FOIDL learnt the past tense for English verb (Mooney and Califf, 1996) (Sameh et al., 1999). For the past tense, the dataset was small. However, one of the major problem of ILP is dealing with large dataset. This is a current research area. A different work using logic programming is the one by Abramson (1992) where Abramson described the relation between the lexical and surface levels of a language using Prolog.

5 Results

In order to test MAGIP, we used the Morpho Challenge dataset of 2010 which contains English, Finnish and Turkish. We used a training set of 1000 morpheme sequences to learn a morpheme graph for every language. For a test set, we used 2928030 words for Finnish, 617298 words for Turkish and 878036 words for English. We selected 2 parses for each word of the test set. Then, we split each test set of parses into subsets of 10000 words. To select a stable model, we generated parses of words from a development set in which we know the morpheme sequences. The development set was of size 694, 763 and 835 for English, Turkish and Finnish respectively. Therefore, we could use the F-measure locally of those subsets, from this we selected a stable model with the highest F-measure among the first thousand stable models if there exists. The results are detailed for English, Turkish and Finnish in the (Table 5). As seen, the F-measure is quite similar among the three languages (.3751 in average). Interestingly, the recall is high with more than 50%. For the Morpho Challenge, the evaluation script needed the label of each morpheme, we decided to label the morpheme by itself.

Language	Precision	Recall	F-Measure
English	.2408	.7133	.3601
Finnish	.2837	.5344	.3706
Turkish	.2776	.6829	.3947

Table 1: Results from the Morpho Challenge 2010.

6 Conclusion

We presented MAGIP (Morphological Analysis by morpheme Graph and mixed-Integer Programming), an algorithm which decomposes words into morphemes for any language. As presented, MAGIP works in two steps, selecting parses and then computing stable models. Our results showed .3751 of F-measure in average for three languages (English, Turkish and Finnish). For future work, we are going to improve the selection of parses by including the notion of frequency in the neighborhood of $MGraph$. Then, we intend to study the size of the mixed-integer program used. Also, we are going to explore others morphological problems where the computation of stable models can be applied.

Acknowledgments

We would like to thank our team colleagues Roger Tucker and Ksenia Shalonova for consultations on general morphological analysis. We would like to thank as well Mr Callum Wright for installing the softwares SCIP and CLP which has allowed us to do the experiments. The work was sponsored by EPSRC grant EP/E010857/1 *Learning the morphology of complex synthetic languages* and the Exabyte Informatics research theme using the BlueCrystal HPC facilities of the Advanced Computing Research Centre, University of Bristol – http://www.bris.ac.uk/acrc/.

References

H. Abramson. 1992. A logic programming view of relational morphology. In *Proceedings of the 14th conference on Computational linguistics*, pages 850–854, Morristown, NJ, USA. Association for Computational Linguistics.

C. Bell, A. Nerode, R. Ng and V. Subrahmanian. 1994. Mixed Integer Programming Methods for Computing Nonmonotonic Deductive Databases.

B. Golénia, S. Spiegler, and P. Flach. 2010. Unsupervised morpheme discovery with ungrade. In *Multilingual Information Access Evaluation*.

K. Clark. 1977. Negation as Failure. In *Logic and Data Bases*, pages 293–322.

M. Gelfond and V. Lifschitz. 1988. The Stable Model Semantics For Logic Programming. In *Proceedings of the Fifth International Conference on Logic Programming (ICLP)*, pages 1070–1080. MIT Press.

R.J. Mooney and M.E. Califf. 1996. Learning the Past Tense of English Verbs Using Inductive Logic Programming. *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing*, pages 370–384.

S. Muggleton and C. Feng. 1990. Efficient induction of logic programs. In *New Generation Computing*. Academic Press.

J. Quinlan and J. Mostow. 1990. Learning logical definitions from relations. In *Machine Learning*, pages 239–266.

A. Sameh, T. Radi, and R. Mikhail. 1999. Learning the past tense of english verbs: An extension to foidl. In *FLAIRS Conference*, pages 298–302.

K. Shalonova and B. Golénia. 2010. Weakly Supervised Morphology Learning for Agglutinating Languages Using Small Training Sets. In *Proceedings of the 23rd International Conference on Computational LINguistics (COLING)*.

Semi-supervised extensions to Morfessor Baseline

Oskar Kohonen and **Sami Virpioja** and **Laura Leppänen** and **Krista Lagus**
Aalto University School of Science and Technology
Adaptive Informatics Research Centre
P.O. Box 15400, FI-00076 AALTO, Finland
firstname.lastname@tkk.fi

Abstract

We have extended Morfessor Baseline, which is a well-known method for unsupervised morphological segmentation, to semi-supervised learning. As submission to Morpho Challenge 2010, we provide results from three methods: The first one is based on the unsupervised algorithm, but includes a weight parameter that can be used to control the amount of segmentation. The second one applies the semi-supervised extension, where the labeled training data is used also during the learning. The third one is based on the second, but as an additional step we label the segments using a Hidden Markov Model trained on the labeled data.

1 Introduction

This work is based on Kohonen et al. (2010), where the Morfessor Baseline method (Creutz and Lagus, 2002; Creutz and Lagus, 2005; Creutz and Lagus, 2007) was extended to the semi-supervised case. Morfessor is a family of generative probabilistic models designed for modeling highly-inflecting and compounding languages (Creutz and Lagus, 2007). It induces a lexicon of word segments, called morphs, from the data. In the semi-supervised version, the training data contains labeled words with known gold standard segmentations. The lexicons that include those segments are favored if the words are added to the data likelihood function. In addition, a small set of word forms with gold standard analyzes can be used for tuning the respective weights of the annotated and unannotated data.

Kohonen et al. (2010) made also a simple experiment on labeling the segmentations provided by the Morfessor to the morpheme labels given in the training data. The results were encouraging considering the trivial labeling method. Here, we extend this line of research by training Hidden Markov Models (HMM) suitable for the task. This results not only in segmentation, but a full morphological analysis of the words.

2 Semi-supervised Morfessor Baseline

Let $\boldsymbol{\theta}$ be the parameters of the model, $\boldsymbol{D}_W$ be the set of word forms used for training the model and $\boldsymbol{D}_{W \to A}$ be a subset of words for which we know the correct morphs. Each word w_j in $\boldsymbol{D}_W$ has a corresponding variable Z_j that denotes how it is segmented. That is, its value is a sequence of morphs, $z_j = (m_{j1}, \ldots, m_{j|z_j|})$. The set of Z_j:s, $\boldsymbol{Z} = (Z_1, \ldots, Z_{|\boldsymbol{D}_W|})$ is a hidden variable that we want to estimate.

A generative model gives the joint distribution $P(W = w, Z = z \,|\, \boldsymbol{\theta})$ of words and their segmentations. Assuming that the sequence of morphs in z can produce only one word type, the probability is simply $P(Z = z \,|\, \boldsymbol{\theta})$ for that word, and zero otherwise. Instead of determining a *posteriori* probability distribution $P(\boldsymbol{\theta}|\boldsymbol{D}_W, \boldsymbol{D}_{W \to A})$ over model parameters $\boldsymbol{\theta}$ as in Bayesian modeling, we try to find a point estimate of $\boldsymbol{\theta}$ given a cost function to minimize. The three main aspects in this framework are:

- What is the family of the model, i.e., how probabilities $P(Z = z \,|\, \boldsymbol{\theta})$ and $P(\boldsymbol{\theta})$ are defined?
- What is the cost function to minimize for selecting $\boldsymbol{\theta}$?
- How to minimize the cost function, i.e., what is the training algorithm?

Next, we shortly describe the applied solution for each of them. Only the cost function differs from the unsupervised Morfessor Baseline.

2.1 Model family

The model family in Morfessor Baseline is relatively simple: The model parameters $\boldsymbol{\theta}$ encode a morph lexicon, which includes the properties of the morphs. Each morph m in the lexicon has a probability of occurring in a word, $P(M = m \,|\, \boldsymbol{\theta})$, and these probabilities are assumed to be independent.

During training, each word w_j is assumed to have only one possible analysis. Thus, instead of using the joint distribution $P(\boldsymbol{D}_W, \boldsymbol{Z} \,|\, \boldsymbol{\theta})$, we need to use the likelihood function only conditioned on the analyses of the observed words, $P(\boldsymbol{D}_W \mid \boldsymbol{Z}, \boldsymbol{\theta})$. The conditional likelihood is

$$\begin{aligned} & P(\boldsymbol{D}_W \,|\, \boldsymbol{Z} = \boldsymbol{z}, \boldsymbol{\theta}) \\ & = \prod_{j=1}^{|\boldsymbol{D}_W|} P(W = w_j \,|\, \boldsymbol{Z} = \boldsymbol{z}, \boldsymbol{\theta}) \\ & = \prod_{j=1}^{|\boldsymbol{D}_W|} \prod_{i=1}^{|z_j|} P(M = m_{ji} \,|\, \boldsymbol{\theta}), \end{aligned} \tag{1}$$

where m_{ij} is the i:th morph in word w_j.

The problem of using Equation 1 for the known segmentations in $\boldsymbol{D}_{W \to A}$ is that there can be alternative segmentations for each word. As a solution, we select only the segmentation that has the highest probability according to the model, and discard the others from the likelihood function. Due to practical reasons, the selection is done only after each training epoch (see Sec. 2.3).

The parameters $\boldsymbol{\theta}$ of the model are:

- Morph type count, or the size of the morph lexicon, $\mu \in \mathbb{Z}_+$
- Morph token count, or the number of morphs tokens in the observed data, $\nu \in \mathbb{Z}_+$
- Morph strings $(\sigma_1, \ldots, \sigma_\mu)$, $\sigma_i \in \Sigma^*$
- Morph counts $(\tau_1, \ldots, \tau_\mu)$, $\tau_i \in \{1, \ldots, \nu\}$, $\sum_i \tau_i = \nu$. Normalized with ν, these give the probabilities of the morphs.

In principle, each parameter has a prior probability. However, with MDL-inspired and non-informative priors, morph type count and morph token counts can be neglected as insignificant. The morph string prior is based on length distribution $P(L)$ and distribution $P(C)$ of characters over the character set Σ, both assumed to be known:

$$P(\sigma_i) = P(L = |\sigma_i|) \prod_{j=1}^{|\sigma_i|} P(C = \sigma_{ij}) \tag{2}$$

We applied the implicit length prior (Creutz and Lagus, 2005), where instead of determining $P(L)$, an end-of-word symbol is used as an additional character in $P(C)$. For morph counts, we used the non-informative prior

$$P(\tau_1, \ldots, \tau_\mu) = 1 / \binom{\nu - 1}{\mu - 1} \tag{3}$$

that gives equal probability to each possible combination of the counts when μ and ν are known.

2.2 Cost function

The unsupervised Morfessor algorithms try to find the maximum a posteriori estimate of the parameters. The equivalent cost function to minimize is

$$L(\boldsymbol{\theta}, \boldsymbol{z}, \boldsymbol{D}_W) = -\ln P(\boldsymbol{\theta}) - \ln P(\boldsymbol{D}_W \,|\, \boldsymbol{z}, \boldsymbol{\theta}). \tag{4}$$

In the semi-supervised version, we add the negative log-likelihood of the known segmentations in $\boldsymbol{D}_{W \to A}$. Furthermore, we weight the data likelihoods with parameters $\alpha > 0$ and $\beta > 0$:

$$\begin{aligned} L(\boldsymbol{\theta}, \boldsymbol{z}, \boldsymbol{D}_W, \boldsymbol{D}_{W \mapsto A}) = \\ & - \ln P(\boldsymbol{\theta}) \\ & - \alpha \times \ln P(\boldsymbol{D}_W \,|\, \boldsymbol{z}, \boldsymbol{\theta}) \\ & - \beta \times \ln P(\boldsymbol{D}_{W \mapsto A} \,|\, \boldsymbol{z}, \boldsymbol{\theta}) \end{aligned} \tag{5}$$

The data likelihood weights control both the level of segmentation, as increasing the weight has to be compensated by a larger morph lexicon, and how large an effect the known segmentations have compared to the unsupervised segmentations.

2.3 Training algorithm

The training algorithm of Morfessor Baseline (Creutz and Lagus, 2005) tries to minimize the cost function by testing local changes to $\boldsymbol{z}$, modifying the parameters according to each change, and selecting the best one. The training algorithm is directly applicable to the semi-supervised case.

The initial parameters are obtained by adding all the words into the morph lexicon. Then, one word is processed at a time, and the segmentation that minimizes the cost function with the optimal model parameters is selected and the parameters are updated respectively:

$$z_j^{(t+1)} = \arg\min_{z_j} \left\{ \min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}, \boldsymbol{z}^{(t)}, \boldsymbol{D}_W) \right\} \quad (6)$$

$$\boldsymbol{\theta}^{(t+1)} = \arg\min_{\boldsymbol{\theta}} \left\{ L(\boldsymbol{\theta}, \boldsymbol{z}^{(t+1)}, \boldsymbol{D}_W) \right\} \quad (7)$$

Because a probability of a morph does not depend on its context, the segmentations in $\boldsymbol{z}$ can be encoded as a tree-like graph, where the words are the top nodes and morphs the leaf nodes. In one training epoch, each top node is processed once. A node can either be left as it is or split into two parts. If the case of a split, the same test is applied recursively to its parts. As the changes cannot increase the cost function, the parameters will converge to a local optimum. In practice, the training is stopped when the average change in cost function per word in an epoch is smaller than 0.005.

3 Morpheme labeling

We use a first-order Hidden Markov Model (HMM) to label the induced morphs (segments of words) to morphemes. The unobserved states are the morpheme labels, and the observations are the segments. We construct the emission alphabet Σ by picking out all the morphs from both the training set and the segmented data that is to be labeled. The set of possible labels (states) is collected from the training data. When the training set does not provide labels for some morphs—as is the case for a large part of the morphs found in the Turkish training set—we group these morphs together under a separate label.

Labels of non-observable morphs, such as the plural morph in the word “men”, are combined with the label of the preceding morph to create a compound label. In the case of the word “men” the compound label would be N+PL. Such compound labels are separated as post-processing. The resulting labeling would thus be “men_N +PL”. Non-observable morphs that start a word are ignored altogether, since they are usually peculiarities in the gold standard labeling. For example, the English gold standard segmentation for the word “propjet” includes a non-observable prefix “turbo”, which is clearly unnecessary.

Hyphens at the beginning or end of a morph such as the one in “-inspired”, the second morph in a segmentation of the word “abba-inspired”, are removed. I.e. “-inspired” is treated as the same morph as “inspired” without the hyphen. Hyphens that are segmented as morphs of their own are taken into account during the calculation of the Viterbi paths but are left out of the result files. Thus, the segmentation “educator - scientist” becomes “educator_N scientist_N” in the results.

Finally, we handle stem allomorphy by replacing morphs with their respective morphemes when provided by the training set. This is done as post-processing. For example, the segmentation “caricatur ish” becomes “caricature_N ish_s”.

3.1 Transition and emission probabilities

After the sets of emissions and labels are collected, maximum likelihood estimation is applied to calculate state transition and emission probabilities from the training data. The probability of a transition from state l_1 to state l_2 is

$$P(l_2 \,|\, l_1) = \frac{C(l_1, l_2)}{C(l_1)}, \quad (8)$$

where $C(l_1, l_2)$ is the number of times l_2 follows l_1 in the training set and $C(l_2)$ is the total number of occurrences of l_2 in the training set.

Similarly, we can estimate that the probability that state l emits morph m is $C(m, l)/C(l)$, where $C(m, l)$ is the number of times m is tagged with l in the training set. However, to accommodate previously unseen morpheme emissions, we apply smoothing to emission probabilities. Smoothing is applied only for labels that represent open classes of morphs, that is, morph classes that can be expanded with new items. For Finnish and English these are nouns, verbs and adjectives. Because the gold standard does not provide labeling for Turkish nouns, verbs and adjectives, we have used the class of morphs that were unlabeled in the gold standard as the only open class when labeling the Turkish data.

As a smoothing method, we use absolute discounting. That is, we subtract a constant value $\delta = 0.1$ from all emission counts $C(m, l)$ greater than zero, and the remaining probability mass is then divided between the previously unseen emissions. Thus, if $N_0(l)$ is the number of emissions for label l with $C(m, l) = 0$, we get

$$P(m \,|\, l) = \begin{cases} \dfrac{C(m,l) - \delta}{C(l)} & \text{if } C(m,l) > 0 \\ \dfrac{(|\Sigma| - N_0(l))\delta}{N_0(l) C(l)} & \text{otherwise.} \end{cases} \quad (9)$$

4 Experiments

We compare four different variants of the Morfessor Baseline algorithm:

- **Unsupervised (U):** The classic, unsupervised Morfessor baseline.
- **Unsupervised + weighting (U+W):** A development set is used for adjusting the weight of the likelihood α. When $\alpha = 1$, the method is equivalent to the unsupervised baseline.
- **Semi-supervised + weighting (S+W):** The semi-supervised method trained with both annotated and unannotated data. The parameters α and β are optimized using the development set.
- **Semi-supervised + weighting + labeling (S+W+L):** As above, but the obtained morphs are labeled with a HMM tagger trained on the annotated training data.

All variants were trained for English, Finnish, and Turkish. Only the unsupervised models were trained for German, as there was no gold standard segmentations available for it. Only the data sets are from the Morpho Challenge 2010 web site[1] were applied. The provided development sets were used for optimizing α and β. The training sets of gold standard segmentations were used in training the semi-supervised segmentation models and the labeling models.

Table 1 shows the values for the optimal weights α and β that were chosen for different languages using the development set in both unsupervised and semi-supervised cases, as well as the respective results. The unsupervised method with weighting (U+W) results in more balanced precision and recall values than the unsupervised baseline method (U), thus clearly increasing the F-measures. The amount of increase is especially large for Finnish and Turkish languages due to the very low recall of the baseline.

The semi-supervised method (S+W) results in a considerable increase in recall and a somewhat more modest increase in precision for English and Finnish. For Turkish, however, we get the opposite result: a large improvement in precision and a small increase in recall. In both cases, the obtained F-measures are clearly better than the ones obtained with the unsupervised training.

Model	α	β	*P %*	*R %*	*F %*
English					
U	-	-	84.75	44.28	58.17
U+W	0.25	-	67.32	60.73	63.86
S+W	0.5	1000	68.46	70.40	69.42
S+W+L	0.5	1000	73.05	68.12	70.50
Finnish					
U	-	-	84.48	17.45	28.92
U+W	0.01	-	59.26	47.00	52.42
S+W	0.01	2000	63.71	60.25	61.93
S+W+L	0.01	500	65.77	67.07	66.41
German					
U	-	-	70.85	22.32	33.95
U+W	0.05	-	56.40	48.53	52.17
Turkish					
U	-	-	94.18	16.85	28.58
U+W	0.01	-	44.91	47.10	45.98
S+W	0.1	1000	73.07	47.95	57.90
S+W+L	0.005	2500	76.95	60.59	67.80

Table 1: The optimal values for the weights α and β and the respective precision (P), recall (R) and F-measure (F) on the development set.

The morpheme labeling (L) should improve recall by solving allomorphy, i.e., finding common labels for the different surface forms, and precision by disambiguating surface forms of different morphemes. In our experiments, the largest increase in F-measure—nearly 10% absolute—is obtained for Turkish, for which the recall increases considerably. For Finnish, the increase is about 4.5% absolute. Both the Finnish and Turkish data sets include a large number of suffixes that have allomorphy, which explains the large improvements. English benefits less from the labeling, gaining slightly over 1% to the F-measure. The increase in precision is larger than for the other languages, but recall is, in fact, decreased by the labeling.

5 Conclusions

We have presented a semi-supervised extension to the Morfessor Baseline method, which performs morphological segmentation using maximum a posteriori estimation. Using gradually more of the information provided by the annotated data sets, we improve the F-measure results on the development set, e.g., for Finnish, from 29% to 52% by optimizing a weight parameter for the data likelihood, to 61% by using the annotated training data in the likelihood function, and finally to 66% by

[1] `www.cis.hut.fi/morphochallenge2010`

using a Hidden Markov Model to label the segmentations. The method could be improved further by using the HMM probabilities directly when segmenting the words. The downside of our approach is that it requires word annotations with both the segmentations (morphs) and their labels (morphemes).

Acknowledgments

This work was funded by Academy of Finland, Graduate School of Language Technology in Finland, and the 7th Framework Programme of the European Commission through the META-NET project (249119).

References

Mathias Creutz and Krista Lagus. 2002. Unsupervised discovery of morphemes. In *Proceedings of the Workshop on Morphological and Phonological Learning of ACL'02*, pages 21–30, Philadelphia, Pennsylvania, USA.

Mathias Creutz and Krista Lagus. 2005. Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0. Technical Report A81, Publications in Computer and Information Science, Helsinki University of Technology.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing*, 4(1), January.

Oskar Kohonen, Sami Virpioja, and Krista Lagus. 2010. Semi-supervised learning of concatenative morphology. In *Proceedings of the Eleventh Meeting of the ACL Special Interest Group on Computational Phonology and Morphology (SIGMORPHON 2010)*, pages 78–86, Uppsala, Sweden. Association for Computational Linguistics.

Learning from Unseen Data

Constantine Lignos
Department of Computer and Information Science
University of Pennsylvania
lignos@cis.upenn.edu

Abstract

The learner of Lignos et al. (2009) attained excellent performance in English in Morpho Challenge 2009, but its reliance on minimal word pairs in the input to learn which words a rule applies to led to poor performance in other languages. We demonstrate that this learner can perform well across a broader set of languages if it works to infer word forms unseen in the data. We evaluate approaches to compounding and base word inference to accomplish this goal, improving the learner's performance greatly in Turkish and Finnish.

1 Introduction

Data sparsity, as best quantified by Zipf's law, is a defining characteristic of language and its impact is felt in unsupervised morphology learning. The pervasive sparsity both between and within lemmas (Chan, 2008) suggests that learners that rely on identifying paradigms will face significant difficulties while learners that learn independent rules can more reliably succeed with less data.

The learner developed by Chan (2008) and extended by Lignos et al. (2009) embraces sparsity by learning independent morphological rules and using characteristics of the distribution of inflected forms to guide the learner's design. This learner was evaluated in Morpho Challenge 2009 (Kurimo et al., 2009) in English and German, but it was not able to model agglutinative languages as the learner reached the limitations of learning rules by minimal pairs of words present in the corpus. While it effectively avoided the difficulty of learning a paradigm representation, it struggled with the sparsity caused by languages in which words consist of many morphemes and there are rarely corresponding minimal pairs for each one.

We extend that learner in this paper by adding features that allow the learner to infer words not present in the corpus, allowing it to succeed without changes to the core minimal pair-based learning model.

2 The Learning Framework

We present a brief summary of the Base and Transforms model and the core operations of the learning algorithm. For further details, see Lignos et al. (2009) and Chan (2008).

2.1 The Base and Transforms Model

A morphologically derived word is modeled as a base word with an accompanying transform that changes the base to create a derived form. A transform is an orthographic modification made to a base to create a derived form. It is defined by two affixes $(s1, s2)$, where $s1$ is removed from the base before concatenating $s2$. A null suffix is represented as $. A transform also has a corresponding word set, which is the set of base-derived pairs that the transform accounts for.

Word Sets. Each word in the corpus belongs to one of three word sets at any point in execution: Base, Derived, or Unmodeled. The Base set contains the words that are used as bases of learned transforms but are not derived from any other form. The Derived set contains words that are derived forms of learned transforms; these words can also serve as bases for other derived forms. All words begin in the Unmodeled set and are moved into Base or Derived as transforms are learned.

2.2 The Learning Loop

Each iteration, the learner does the following:

1. Counts the affixes that appear in each word set, ignoring low frequency words.

2. Hypothesizes transforms between pairs of the most frequent affixes and scores each transform using the number of word pairs it models and the amount it changes the base word. For example, the transform *($, s)* can model the pair *paper/papers*.

3. Selects the highest scoring transform and moves the words modeled by that transform into the Base and Derived sets as appropriate.

The learning loop continues until none of the highest ranked transforms meet the criteria for an acceptable transform. After learning is complete, each word is analyzed using its base word and any transforms required to derive it from the base.

3 Additions to the Framework

While the innovations introduced by Lignos et al. (2009) addressed the largest gaps between the cognitive model proposed by Chan (2008) and the requirements of a morphological analyzer, as shown in the results of Morpho Challenge 2009 (Kurimo et al., 2009) the algorithm's success was primarily limited to English. Modest results were reported in German, but the algorithm was not submitted for other languages because it was unable to handle languages with many morphemes per word. We add the following features to allow the learner to maintain its core learning process while adding additional words to the lexicon as it models the corpus. We show how our features integrate with the algorithm of Lignos et al. (2009) in Figure 1.

3.1 Base Inference

As an example of the limitations of applying rules based on minimal pairs, consider three words appearing in the Brown corpus (Francis and Kucera, 1967): *adjoins, adjoined, adjoining*. Even though the transforms *($, s)*, *($, ed)*, and *($, ing)* are learned, they cannot be used to model these three words because the required base, *adjoin*, is not present in the corpus. This results in lower recall because these three words remain unmodeled despite the fact that the rules required to model them have been learned.

To address situations of this type, we introduce *base inference* to infer the existence of an unseen word when more than one learned transform suggests that it should exist. After each rule is learned, the learner iterates over every word with the transform's $s2$ affix that was not successfully modeled by the transform and notes the base that would have been required to model that word. If a later transform requires the same base to model another word, the learner infers the existence of that base and then attempts to use that base in all transforms learned to that point and those learned later.

For example, consider the learner's operation when learning the transforms *($, s)*, *($, ed)*, and *($, ing)* in order while *adjoins*, *adjoined*, and *adjoining* are present in the corpus but *adjoin* is not. After the transform *($, s)* is learned, the learner iterates over all words containing the suffix *-s* that were not modeled, including *adjoins*, and notes the required base, *adjoin*. In the next iteration, the learner selects the transform *($, ed)* and similarly notes that the required base for *adjoined* is *adjoin*. Because two different rules have indicated the possible existence of *adjoin*, the learner infers its existence. The learner adds it to the lexicon with the frequency of the word that caused it to be inferred and marks it for the transforms *($, s)* and *($, ed)*, thus modeling *adjoins* and *adjoined*. In the next iteration, since *adjoin* is now in the Base word set, when the transform *($, ing)* is learned, *adjoining* can be modeled as if *adjoin* were present in the input corpus.

3.2 Compounding

The model presented by Lignos et al. (2009) performs a simple n-gram based compounding as a post-processing step on the learner's output. But as with base inference, it would be beneficial for the algorithm to be able to use the components of a compound word during learning instead of only breaking compounds after analysis.

We adopt the compounding model of Koehn and Knight (2003) where a word is broken down into the set of component words present in the lexicon with the highest geometric mean of frequencies. Thus the splitter selects a split using the following equation for a split S comprised of $w_i \ldots w_n$ component words:

$$\arg\max_S \Big(\prod_{w_i \in S} count(w_i) \Big)^{\frac{1}{n}}$$

The null hypothesis is also considered where S contains only the word being split, thus a word is only split if the geometric mean of its component

```
1. Add all words in the corpus to the Unmodeled set.
2. Until a stopping condition is met, perform the main learning loop:
    (a) Score suffixes and transforms and select the best transform.
    (b) Move the words used in the selected transform.
    (c) Optionally perform Base Inference, inferring new bases and adding them to learned transforms as appropriate.
    (d) Optionally perform compounding for the current iteration.
3. Optionally perform compounding after learning is complete.
```

Figure 1: An overview of the learning algorithm, integrating compounding and inference features

words' frequencies is greater than the word's frequency. Koehn and Knight also use "filler," character sequences that can be placed between component words of a compound. Rather than specifying the filler sequences by hand, we allow the filler to be the application of a previously learned transform to a word in the lexicon when including it in the compound. The learner only allows component words from the Base or Derived sets to have filler added to them; this helps in excluding morphologically unproductive words from having transforms applied to them for the purpose of compounding.

If a compound word is split using a transform applied to a word in the lexicon, the derived form is added to the lexicon and marked as derived from the word the transform was applied to. For example, consider an example from Koehn and Knight (2003) where we are splitting *Aktionsplan*. Assume that *Aktion* and *plan* are in the lexicon but *Aktions* is not and that the learner has learned the transform *($, s)*. Assuming the correct frequency requirements are met, the learner would break the compound as *Aktions* and *plan*, where *Aktions* was derived by applying *($, s)* to *Aktion*. The learner would add *Aktions* to the lexicon, noting the relationship to *Aktion*. This provides the companion to Base Inference for derived words; the learner infers the existence of a derived form by its presence in a compound. The learner is then able to learn words derived from *Aktions* if needed, a crucial ability in agglutinative languages which typically contain many compounds where the component words can take a large number of suffixes but may not be observed elsewhere in the corpus.

We apply the compounding approach in three variants:

Basic Compounding. Compounding is applied to words in the Base and Unmodeled sets after all learning is complete, and no transforms are supplied as fillers for the compounding system.

Iterative Compounding. Compounding is applied to words in the Base set after every iteration and to the Unmodeled set after all learning is complete. The transforms learned up to the current iteration are always supplied as fillers for the compounding system.

Aggressive Compounding. Compounding is applied to words in the Base and Unmodeled sets after every iteration. As in Iterative Compounding, the transforms learned up to the current iteration are always supplied as fillers for the compounding system, but the key difference is that they are applied to the words in Unmodeled every iteration, not just when learning is complete. This is more aggressive because it introduces many more words during the learning process than if unmodeled words are only split after learning is complete.

4 Results

The learner's performance on the development set of Morpho Challenge 2010 is given in Table 1. The *Base* condition gives the performance of the learner without any compounding or word inference features active. The *Basic Compounding* condition gives the performance of the learner with Basic Compounding in use but without Base Inference. The *Base Inference* condition builds on the Basic Compounding condition, adding the Base Inference feature. The *Iterative Compounding* and *Aggressive Compounding* conditions build on the Base Inference condition, with their forms of compounding superseding Basic Compounding.

The results show that while the features introduced lead to mixed results on the F-score for En-

	Precision	Recall	F-score
English			
Base	60.56	50.47	55.05
+Basic Compounding	59.26	52.82	55.85
+Base Inference	59.26	54.21	**56.62**
+Iter. Compounding	57.80	52.07	54.79
+Aggr. Compounding	46.57	51.81	49.05
Finnish			
Base	69.19	09.89	17.30
+Basic Compounding	65.55	26.32	37.55
+Base Inference	76.40	26.84	39.72
+Iter. Compounding	72.85	29.32	41.81
+Aggr. Compounding	54.36	44.49	**48.93**
German			
Base	57.72	28.03	37.73
+Basic Compounding	42.17	34.08	37.70
+Base Inference	44.74	34.22	38.78
+Iter. Compounding	46.69	32.78	**38.52**
+Aggr. Compounding	38.52	35.02	36.69
Turkish			
Base	70.00	9.73	17.08
+Basic Compounding	61.68	12.78	21.17
+Base Inference	50.33	13.74	21.59
+Iter. Compounding	49.45	19.56	28.03
+Aggr. Compounding	35.19	31.63	**33.31**

Table 1: Learner performance on the Morpho Challenge 2010 development sets

glish and German, they improve F-score greatly in Turkish and Finnish. The Basic Compounding feature results in little change in English and German but moderate improvement in Turkish and a dramatic improvement in Finnish. Both Turkish and Finnish benefit especially from Aggressive Compounding, but while German sees a very small performance drop as compared to Iterative Compounding, the drop in English precision and F-score is large.

While in all languages the Base Inference feature led to a small gain in F-score, the impact of the feature was less than was expected. The likely cause is that the conditions that motivate the base inference feature are rare; the algorithm's design leads to the base words of a transform being more frequent on average than the derived words, so Base Inference only handles a small number of exceptions that are unlikely to be evaluated by the small development set. We expected Base Inference to provide a gain in recall with little impact to precision, but the Finnish and German results puzzlingly show precision improvements with almost no recall improvements.

Based on these results, we are submitting the analyses of the Base Inference, Iterative Compounding, and Aggressive Compounding conditions to be evaluated in Morpho Challenge 2010.

5 Discussion

The features present here succeed in transforming the learner into one better suited for agglutinative languages. The strongest improvement in agglutinative languages, however, comes at the expense of precision, most notably when Aggressive Compounding is used. Ideally, a single technique would result in the greatest performance across all languages, but as evaluated with the Morpho Challenge 2010 development set Aggressive Compounding results in the best results for Turkish and Finnish, near-best results for German, and the worst results for English.

As the name implies, Aggressive Compounding applies any transform learned to a word in the Base or Derived sets without any further criteria or any penalty for applying the transform. Given the significant drops in precision caused by using Aggressive Compounding, it appears that a restriction or regularization of some form is required. This points out a fundamental shortcoming of this learner: the learner does not understand the condition for applying a transform beyond suffixes on the base word. Learning part of speech information would likely help the learner decide whether a word can take a transform or not as a word's part of speech can determine what morphemes can be used with it.

The improvements to the learner for Morpho Challenge 2010 further support our position that a non-statistical approach to morphology learning can succeed in a variety of languages.

References

E. Chan. 2008. *Structures and distributions in morphology learning*. Ph.D. thesis, University of Pennsylvania.

S. Francis and H. Kucera. 1967. Computing analysis of present-day American English.

P. Koehn and K. Knight. 2003. Empirical methods for compound splitting. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 187–193. Association for Computational Linguistics.

Mikko Kurimo, Sami Virpioja, Ville T. Turunen, Graeme W. Blackwood, and William Byrneg. 2009. Overview and results of Morpho Challenge 2009. In *Working Notes of the 10th Workshop of the Cross-Language Evaluation Forum*, Corfu, Greece, September 30–October 2. CLEF2009.

Constantine Lignos, Erwin Chan, Mitchell P. Marcus, and Charles Yang. 2009. A Rule-Based Unsupervised Morphology Learning Framework. In *Working Notes of the 10th Workshop of the Cross-Language Evaluation Forum*, Corfu, Greece, September 30–October 2. CLEF2009.

Unsupervised learning of concatenative morphology based on frequency-related form occurrence

Lionel Nicolas
Team RL, Laboratory I3S
UNSA + CNRS,
Sophia Antipolis, France
lnicolas@i3s.unice.fr

Jacques Farré
Team RL, Laboratory I3S
UNSA + CNRS,
Sophia Antipolis, France
Jacques.Farre@unice.fr

Miguel A. Molinero
Grupo LYS,
Univ. de A Coruña,
A Coruña, España
mmolinero@udc.es

Abstract

We describe an unsupervised approach to acquire from raw data a data-orientated representation of a concatenative morphology. This approach takes advantage of various phenomena and, among them, it exploits the fact that the more frequent a form is, the more chances there are to find its morphologically related forms.

Since its implementation is based on straightforward statements and rather simple computations, its efficiency tend to rely more on the size of the input corpus and the adequation between the statements and the language studied than it relies on the intern formulas or parameters. The expert work required to apply it to various languages is thus greatly reduced.

1 Introduction

Among the numeric linguistic resources formalizing a language (grammar, lexicon, etc.), morphological rules are considered as one of the easiest to create. Nevertheless, the construction of such resource still requires a lot of linguistic expertise and the fact is that, for many major languages, such resources are not always available and, for languages with smaller communities, they barely exist. Therefore, the automatized acquisition of morphology is still an open topic which interest is confirmed by its dedicated annual challenge (Mikko Kurimo and Turunen, 2009).

In this abstract, we introduce an approach to compute, from a raw corpora, a data-orientated description of the concatenative morphology used in the input corpora. This approach presents the benefit of taking advantage of phenomena that are observable for concatenative morphologies. Among them, a frequency-related occurrence of the forms belonging to a same lemma, be it derivated or inflected, is highlighted and intensively exploited in a way that has not been considered so far.

The implementation works as a set of filters that refines sequentially a list of candidate affixes and a list of sets of affixes. Since these filters are implemented with mostly straightforward and parameters-free formulas, applying this approach to a varied set of concatenative languages is achieved with few expert work.

In sect. 2, we introduce general definitions and informations while in sect. 3, we discuss the frequency-related phenomenon. In sect. 4, we expose other techniques available. We then describe, in sect. 5, 6 and 7, the various steps of our approach and finally, we expose the data submission details (sect. 8) and conclude (sect. 9).

2 General definitions and informations

General definitions So as to ensure a good understanding, central notions, as we understand and use them, are briefly reminded. *A lemma* is a set of morphologically related forms[1]. We consider *the stem* of a form as the substring shared by *all the related forms of its lemma*. *Affixes* are the substrings added to the stems in order to create forms. *Affix* is here used as a shortcut for *prefix* and *suffix*. *Morphological rules* are string operations converting a stem or a canonical form into a related lexical form. In this paper, a morphological rule consists in adding an affix to a given stem with no character deletion or substitution. Linguistic phenomena which can alter the stem or the affix are acquired as different morphological rules than the ones they are originally derivated from. *A morphological family* is a set of *morphological rules*.

General information Some of the computations described are done thanks to "letter trees", i.e, trees with letter-labeled branches, where every

[1] Each lemma is designed by a "canonical" form;

form is introduced letter by letter and spread over the different sub-parts. When looking for suffixes, a tree is built by introducing the forms from their first letter and oppositely when looking for prefixes. An affix is said *to occur on a given node* if it has been combined with a stem which letters label a path from the root to this node[2]. *A frequent form* is a form which frequency is above the average frequency computed over all the forms of the input corpus. Finally, even thought we *abstractly* refer to the lemmas of the forms, we do not know them since only raw text is provided as input.

3 Frequency-related phenomenon

Since their semantic meaning, be it general or not, matches better the content of some texts, some lemmas are more frequent than others. When considering a given lemma, the probabilities for its inflected or derivated forms to occur in a text increase with the frequency of the lemma. In other words, the more a lemma is "used" in a corpus, the more probable it is to encounter a more diversified sample of its related forms. This applies to most kind of text[3], be it specialized or general, except those describing exhaustively the lexical forms related to some particular lemmas. Of course, the style affects the ratio between related forms[4], but the chances for related forms to occur still globally increase with the frequency of the lemma.

4 Related work

In this section, we focus on approaches related to ours. Techniques inspired by (Harris, 1954) measure the probabilities of word-internal character transitions to identify morpheme boundaries (Hafer and Weiss, 1974; Déjean, 1998; Demberg, 2007). Some of them also use letter-trees, but their computations are done over the nodes when ours are done over substrings occurring on a node, i.e, we only use trees as convenient data-representations to know which forms share substrings. Other approaches rely on the minimum description length (MDL) principle to exhibit a set of descriptive morphemes in order to efficiently encode a corpus (Goldsmith, 2001; Creutz and Lagus, 2002; Snover and Brent, 2002; Bernhard, 2006; Monson et al., 2007). In particular, Goldsmith groups morphemes into paradigm-like sets of suffixes which are close to our morphological families (see sect. 6). Techniques relying on formal analogy (Stroppa and Yvon, 2005; Hathout, 2008; Lavallée and Langlais, 2009) identifies pairs of morphologically related form that are comparable with our pairs of candidate affixes (see sect. 5.1). At last, the approach described in (Dasgupta and Ng, 2007) is similar to ours since it sequentially refines a list of candidate affixes.

Most of these existing approaches share similarities with ours. Nevertheless, the main differences can be globally summarized as: (1) it sequentially applies several complementary strategies while the others mostly focus on one or two, (2) it takes advantage of a frequency-related phenomenon in a fashion that has not been considered before, (3) it is developed towards the aim of avoiding parameters or sophisticated formulas in order to ease its application to other languages.

5 Identifying candidate affixes

Generating candidate affixes This step aims at establishing a rough list of candidate affixes for the later steps that are more computationally-intensive. This first list is obtained by identifying the substrings fulfilling these voluntarily "permissive" conditions: (1) it occurs in at least one subtree that is present at least twice in the whole tree, (2) it occurs in at least one node covering a frequent word, (3) it is globally combined with more substrings than the various "candidate" stems it is combined with, (4) it occurs more frequently on nodes with other substrings than it occurs alone, and finally, (5) it co-occurs with at least another candidate affix on at worst two nodes.

5.1 Identifying pairs of candidate affixes

This step relies on two observations. First, a morphological family covers at least two affixes[5]. The second is based on the frequency-related phenomenon (see sect. 3). Contrarily to other kind of morphology which can use infixes, concatenative morphologies do not alter the stems. Thus, the forms of a lemma follow a same path in the tree until they pass the last letter of the stem. Therefore, in order to find the related candidate affixes of a candidate affix *aff*, one only needs to pay attention to the nodes where *aff* occurs.

[2] An affix combined with n stems will occur on n nodes.

[3] e.g, the forms of *to talk* are easier to encounter in general texts than the ones of *to orate*.

[4] E.g, an autobiography enhance the first person singular.

[5] The empty string is considered as an affix

If the candidate affix studied truly belongs to a family, the more frequent the form containing it is, the more probable the affix occurs on the corresponding node with other affixes of the family. Thus, by sorting by frequency the nodes where a correct candidate affix occurs, we observe a progressively increasing co-occurrence rate with other members of the family. Oppositely, incorrect candidate affixes have chaotic/non-coherent co-occurrence rates with other candidate affixes.

In order to establish if a candidate affix *aff* presents an increasing co-occurrence rate with another one, we divide the sorted list of the nodes where *aff* occurs in sublists for which the average frequency of a sublist s_i is $mult$ times higher ($mult > 1$) than the previous sublist s_{i-1}[6]. We then compute for each candidate affixes co-appearing with *aff* a co-occurrence rate $rate_i$ over each sublist and a score $inc = sum_{pos} + mult * sum_{neg}$ where sum_{pos} is the sum of the positive value $rate_i - rate_{i-1}$, whereas sum_{neg} is the sum of the negative ones. The co-occurrence is considered as increasing if inc is positive. Candidates with no increasing co-occurrence are discarded.

One must note that this filter identifies pairs of incorrect candidate affixes because of correct ones. Indeed, any affix *X* related to an affix *Y* often allows the candidate affixes *subX* and *subY* starting with a string *sub* to be considered as related since their co-appearance rate will also be an increasing one[7]. The same applies for two incorrect affixes *Z* and *W* and two correct and related affixes *subZ* and *subW* that start with a string *sub*[8].

6 Morphological families

6.1 Building morphological families

Once pairs are identified, we recursively process the tree with the remaining candidates. Every candidate affix occurring on a given node "vote" for its pairs also present on the node. A family is then built with the pairs of the candidate affix that has received most votes. The voting is done several times with the candidates non-included in a family until all are included in one[9]. For each family, we record the nodes where they have been found.

6.2 Filtering morphological families

The previous step generates three kinds of families: some incorrect ones composed of incorrect affixes, correct but incomplete ones and correct and complete ones. The first set is mainly composed composed of families which members englobe correct affixes (see end of sect. 5.1). The second set is essentially composed of correct subfamilies of bigger families: as explained in sect. 3, depending on the lemma, more or less related forms are found in the input corpus and thus, more or less complete families are generated.

We thus apply sequentially three filters to the families. The first aims at discarding the biggest incorrect families and most of the incomplete ones. It relies on the following idea: a correct family composed of *n* members shall appear in subfamilies with *n*, *n-1*, *n-2*,...,*1* of its members (see sect. 3). A family with *n* members is thus kept if "validated" by the occurrence of at least one family with *n-1* of its members[10]. All families validating another one are discarded (essentially subfamilies) as well as families that have not been validated (essentially the biggest incorrect families). One must notice that if two equivalent families with *n+1* members sharing *n* of their members are generated, this filter will keep both unless a family with *n+2* members covering them appears. The second filter relies on the idea that morphological families are frequency-independent, i.e, they apply on frequent or infrequent lemmas and should thus cover, in one of the nodes where they have been found, at least one of the frequent forms. The third filter follows the idea that the letters common to all related forms belong to the stem (see sect. 2) and tackles the incorrect families generated by pairs of incorrect affixes englobing correct ones (like "med#/ming#", see end of sect. 5.1) by simply rejecting all families composed of affixes starting with an unique first letter.

6.3 Splitting compound affixes

If an affix *aff3* in a family *fam1* is to be split as two affixes *aff1*, *aff2*[11], we consider that *aff3* is obtained by refining an affix *aff1* with a family

[6] The first sublist is the set of nodes corresponding to the forms with the lowest frequency.

[7] E.g, the english suffixes "*ing#*"/"*ed#*" can allow the incorrect ones "*ming#*"/"*med#*" to be considered as related.

[8] E.g, the pair of english suffixes "*es#*"/"*ed#*" can allow the pair "*s#*"/"*d#*" to be identified.

[9] This method has been designed to avoid merging together two different families present on a same node. E.g, the spanish verbs "sentir"(*to feel*) and "sentar"(*to sit*) belong to two different families but share the same stem "sent".

[10] Families with two members are automatically validated.

[11] e.g, the english suffix "ingly#" is to be split as the suffixes "ing#" and "ly#"

fam2 containing the affix *aff2*. We thus consider that *aff1* is the reason for *aff3* to exist and that *aff1* provides of a context where *fam2* can apply, i.e, there can be other affixes in the family *fam1* obtained by refining *aff1* with *fam2*. We thus list the other affixes in *fam* that could be reason for *aff3* to exist by following the idea that they should be more present than absent on the nodes where *aff3* occurs. We then add *aff3* to the list and apply the families obtained in the previous step. The family that covers most of the elements of this list, including *aff3*, is selected, *aff1* is split as *aff1*+*aff2* and the process is recursively applied on *aff2*.

7 Cutting forms

Once the final set of families is obtained, we use them to split every word as *prefix(es) + stem + suffix(es)*. This step is achieved by selecting for each nodes the families that apply. A family is said to apply on a node if it covers *n* ($n > 2$) substrings occurring on the node and thus generate a set of *n* possible cuts of the corresponding forms.

For each form, a choice among the various sets of cuts covering it is achieved by selecting them sequentially according to the following two criteria: (1) the number of cuts of the set, (2) the corresponding node closest to the root (3) the size of the corresponding family[12]. If more than one set remains[13], we simply select the first one.

7.1 Splitting compound stems

So as to split compound stems, one needs first to determine what substrings can connect them, be it the empty string or not. During our experiments, we could observe that these *connectors* act like double-affixes in the sense that they tend to combine two surrounding stems the same way suffixes are connected to the first stem and prefixes are connected to the second one. We also observed that, if enough data is provided, the most frequent words tend to be identified along with connectors in ”fake“ suffixes or prefixes[14]. In order to extract these connectors, we first establish two lists of *starting* and *ending* substrings corresponding to the combination of all the stems found previously (see sect. 7) with the prefixes and suffixes they have been found with[15]. We then identify all the prefixes containing *starting* substrings and all the suffixes containing *ending* substrings. The substring part of these ”fake“ affixes that do not belong to the *starting* or *ending* substrings are considered as possible connectors. A connector is kept if it is both found in one ”fake“ prefix and one ”fake“ suffix[16]. Finally, the stem of a given form is recursively split if the form combines a *starting* substring, a connector and an *ending* substring[17].

8 Data submission

The morphological analysis achieved for English, Turkish and German, have been produced by giving as input the basic corpora provided by the 2010 edition without any parameters or code adaptations[18]. The process just differs in the size of the input corpus. Especially since we limited, because of a memory issue, the input to the forms with a frequency greater than 1 for English and 2 for German. For Finnish, we did not provide analysis because the date processed was not enough data to achieve an interesting result.

9 Partial Conclusion

The approach alredy achieves its original goal by generating a set of the morphological families present in the input corpus.

The preliminary results indicates a generally good precision and a recall that lowers with the complexity of the morphology. Nevertheless, this drawback decreases when the size of the input corpus increases. We also observe, from our analysis over English, that an important loss in both recall and precision is due to allomorphy. Indeed, we are unable to recognize syntactically equivalent affixes (loss of recall), or differentiate a same affix shared by two syntactically non-equivalent families (loss in precision).

Finally, the results obtained over various languages with no parameter or code adaptations tend to indicate the relevance and interest of the approach.

[12] the biggest families are the most correct ones

[13] some families can be sub-families of a bigger non-detected one and thus compete often

[14] e.g, in English, the prefixes ”#grand“, ”#first-“ and the suffixes ”-based#“, ”ship#“ are identified.

[15] e.g, if the english stem ”appear“ is found with the prefixes ”#re“ and ”#dis“ and the suffixes ”ing#“ and ”ed#“, the substrings ”#reappear“ and ”#disappear“ are used as *starting* ones and ”appearing#“ and ’appeared#“ as *ending* ones.

[16] e.g, in English, the connector ”-“ is found in the ”fake“ prefix ”#first-“ and in the ”fake“ suffix ”-based#“.

[17] e.g, ”speedboat“ has the stems of the forms ”speed“ and ”boat“ since it combines a *starting* substring ”#speed“, an empty connector ” and an *ending* substring “boat#”.

[18] The parameter $mult$ (see sect. 5.1) was set to 2.

References

Delphine Bernhard. 2006. Automatic acquisition of semantic relationships from morphological relatedness. In *Advances in Natural Language Processing, 5th International Conference on NLP, FinTAL 2006*, volume 4139 of *Lecture Notes in Computer Science*, pages 121–132. Springer.

Mathias Creutz and Krista Lagus. 2002. Unsupervised discovery of morphemes. In *Workshop on Morphological and Phonological Learning of ACL-02*, pages 21–30.

Sajib Dasgupta and Vincent Ng. 2007. High-performance, language-independent morphological segmentation. In *NAACL HLT 2007: Proceedings of the Main Conference*, pages 155–163.

Hervé Déjean. 1998. Morphemes as necessary concept for structures discovery from untagged corpora. In *NeMLaP3/CoNLL '98: Proceedings of the Joint Conferences on New Methods in Language Processing and Computational Natural Language Learning*, pages 295–298. The Association for Computational Linguistics.

Vera Demberg. 2007. A language-independent unsupervised model for morphological segmentation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*. The Association for Computer Linguistics.

John Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27(2):153–198.

Margaret A. Hafer and Stephen F. Weiss. 1974. Word segmentation by letter successor varieties. *Information Storage and Retrieval*, 10(11-12):371–385.

Zellig S. Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

Nabil Hathout. 2008. Acquistion of the morphological structure of the lexicon based on lexical similarity and formal analogy. In *TextGraphs '08: Proceedings of the 3rd Textgraphs Workshop on Graph-Based Algorithms for Natural Language Processing*, pages 1–8. The Association for Computational Linguistics.

Jean-François Lavallée and Philippe Langlais. 2009. Morphological acquisition by formal analogy. In *Morpho Challenge Workshop 2009*. www.cis.hut.fi/morphochallenge2009.

Sami Virpioja Mikko Kurimo and Ville Turunen. 2009. Unsupervised morpheme analysis – morpho challenge 2009. www.cis.hut.fi/morphochallenge2009.

Christian Monson, Jaime G. Carbonell, Alon Lavie, and Lori S. Levin. 2007. Paramor: Finding paradigms across morphology. In *CLEF*, volume 5152 of *Lecture Notes in Computer Science*, pages 900–907. Springer.

Matthew G. Snover and Michael R. Brent. 2002. A probabilistic model for learning concatenative morphology. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *NIPS*, pages 1513–1520. MIT Press.

Nicolas Stroppa and François Yvon. 2005. An analogical learner for morphological analysis. In *9th Conference on Computational Natural Language Learning (CoNLL 2005)*, pages 120–127.

DEAP: Deductive-abductive parsing for morphological analysis

Sebastian Spiegler, Bruno Golénia, Peter A. Flach
Intelligent Systems Group
University of Bristol, UK
{spiegler, goleniab, flach}@cs.bris.ac.uk

Abstract

In this paper we present a supervised algorithm for deductive-abductive parsing, called DEAP, which generates a set of parse hypotheses for a given word, ranks them probabilistically or by a minimum description length heuristic and selects the top k parses which describe the structure of a word in terms of labelled morphemes. The morphological grammar used by the parser is attained through memory-based learning from a small labelled training set. For the Morpho Challenge 2010, we report results on the languages English, Finnish and Turkish.

1 Introduction

The Morpho Challenge (MC) 2010 is concerned with semi- and supervised morpheme analysis in terms of decomposing words into morphemes, labelling morphemes according to grammatical categories and detecting morphophonological changes. The goal is to develop a machine learning algorithm which can build a morphological model using a small word list of morphologically analysed words and then apply the model to a large unlabelled list of words from the same language to predict their morphological structure.

A *morphological model* is a simplified description of a language's morphology and an instance of a *model class*. The solution at hand is a hybrid of rule-based, probabilistic and statistical models. By exploiting the structure of the labelled training set, where for each word a single or multiple analyses are given as a sequence of morphemes and labels corresponding to a particular morpheme category, we build a *context-free grammar*. This grammar consists of rules describing possible label sequences and a dictionary which maps actual morphemes to their labels or categories.

Such a grammar, which is learnt from a small subset, however, is likely to be incomplete and will not parse all unseen words. For this reason, we apply the notion of *open* and *closed* morpheme categories. The former corresponds to categories whose morphemes cannot be enumerated entirely. Examples are noun, verb and adjective roots or stems. For the latter, in contrast, it is possible to list most of the morphemes since they normally mark certain grammatical properties which themselves are of finite nature.

DEAP, which stands for *de*ductive-*a*bductive *p*arsing, is a further development of the algorithm described by Spiegler et al. (2010) incorporating an automated parse selection. DEAP has been developed under the assumptions that instances of the most important morpheme categories like prefixes and suffixes have been seen in the training set as well as most of the possible combinations of different categories. Only certain morpheme categories, e.g. noun or verb roots, have to be hypothesised. DEAP generates a parse *hypothesis* of a word if either all or all but one morphemes are known. A guessed morpheme must belong to a predefined category whose members are allowed to be inferred. DEAP then performs *hypothesis selection* where it ranks parses probabilistically or by a minimum description length-based heuristic and selects the top k solutions.

In the remainder of the paper, we will present the single steps of the DEAP algorithm and its results on the English, Finnish and Turkish dataset of MC 2010.

2 DEAP: Deductive-abductive parsing

2.1 Datasets and pre-processing

For our approach, we need a training dataset which assigns morphological analyses to each word consisting of morphemes and their categories. This type of dataset was provided for English, Finnish and Turkish. Apart from replacing certain special characters like apostrophes and hyphens by alphanumeric symbols, and changing the decoding of Finnish from 'latin-1' to standard encoding 'ascii', we kept the gold standards as they were.

In contrast, the word lists which had to be analysed underwent a more extensive preprocessing. Since the English training set consisting of 1000 words contained only 42 examples of hyphenated word compounds, but the unlabelled list of almost 900k words more than one third, we decided to process words in these compounds separately and later recombine single analyses again. From the lists of Turkish and Finnish we removed words which did not comply with their original encoding. Furthermore, the entire word list of Finnish was transformed into standard 'ascii' encoding for the subsequent processing steps.

2.2 Memory-based grammar induction

The first step of the DEAP algorithm is the induction of a morphological grammar which will be applied in the hypothesis generation step. We represent the morphological grammar as a *context-free grammar* consisting of rules where a single symbol on the left-hand side is resolved into multiple symbols on the right-hand side. Furthermore, the grammar contains a dictionary where symbols are mapped to actual morphemes.

We are sharing the opinion of Daelemans and Bosch (2005) that generalisation in morphological analysis is harmful since artificial structures can be created easily which do not occur in a natural language. For this reason, we are inducing our morphological grammar using a *memory-based approach*. Morphological analyses of words are directly transformed into rules and dictionary entries. A new rule is a previously unseen sequence of morpheme categories. Each category itself is mapped to its actual morpheme in the dictionary.

Example 1. The analysis of 'deciphering → de:P cipher:N ing:PCP1' would yield the rule '$w \rightarrow p, n, pcp1.$' and dictionary entries '$p \rightarrow [de].$', '$n \rightarrow [cipher].$' and '$pcp1 \rightarrow [ing].$'.

2.3 Grammar augmentation for abduction

Although we assume that the most important combinations of morpheme categories have been seen and therefore transformed into an almost complete rule set, it is unlikely that our morpheme dictionary will be sufficient. We revert to an inference method called *abduction* which is a type of logical inference where an explanatory hypothesis is formed from an observation requiring explanation (Peirce, 1958; Flach and Kakas, 2000). We are trying to morphologically analyse a given word where the most a single morpheme has to be hypothesized from predefined morpheme categories. We refer to these categories as *abducibles*. It is not too difficult to find a set of abducibles for a given language with sufficient linguistic background knowledge. However, we follow a more objective approach which ranks morpheme categories by their number of distinct morphemes. We then choose the top k categories with the highest cardinality since they are likely to be instances of open categories.

Having settled on a list of abducibles, we are now returning to our grammar. So far, the rules can only be applied if all morpheme categories can be mapped to known morphemes. For rules containing categories which are abducible, we want to add modified rule versions to the grammar where the most one abducible is labelled by a marker. This marker allows that a morpheme can be hypothesised for this category. If a rule contains multiple abducibles, combinations with only one marked abducible are added to the grammar.

Example 2. The analysis of the word '*storerooms*' is given as '*store:V room:N s:+PL*' and generates the rule '$w \rightarrow v, n, pl.$'. The morpheme categories '*v*' and '*n*' are listed as abducibles such that the two rules '$w \rightarrow v^*, n, pl.$' and '$w \rightarrow v, n^*, pl.$' will also be added to the rule list. Morphemes for v^* and n^* can be hypothesised.

2.4 Hypothesis generation

For hypothesis generation we apply the formalism of *Definite Clause Grammars (DCGs)* used in the logic programming language *Prolog*. Since we do not only want to accept or reject words based on the grammar but get possible parses of the word instead, we need to augment the DCG by adding parse construction arguments illustrated below.

Example 3. The original DCG may look like:

```
w   --> v, n, pl.
v   --> [store].
n   --> [room].
pl  --> [s].
```

and with parse construction arguments like:

```
w((X, Y, Z)) -->  v(X),n(Y),pl(Z).
```

```
v(v(store))   --> [store].
n(n(room))    --> [room].
pl(pl(s))     --> [s].
```

The word '*storerooms*' would be parsed as:

```
(v(store), n(room), pl(s)).
```

Having defined the grammar with additional rules for abducible morpheme categories and parse construction arguments, we now apply a logic program for the *hypothesis generation* in terms of morphological analyses for given words. The inbuilt Prolog parser only performs *deduction* as logical inference where the hypothesis is formed from observations and known explanations – in our case – a complete rule set and morpheme dictionary. We therefore revert to an extension of the meta-interpreter designed by Flach (1994) which is able to perform deduction as well as abduction. The algorithm invokes rules top-down starting with the most general until it reaches the level where morpheme categories are resolved to actual morphemes. It then matches categories to morphemes from the left to the right of the word. A word is parsed if either all morphemes are listed in the dictionary or unknown morphemes belong to abducible morpheme categories.

2.5 Hypothesis selection

In the previous step, we have enumerated all possible parses for a given word and a grammar. If a word can be parsed *deductively*, we keep all analyses since it is likely that they were caused by the inherent ambiguity due to the lacking word context. On the other hand, if we have *abduced* parses only, we want to select the best hypotheses by evaluating the possible explanations of a word. Subsequently, we present two approaches for abductive hypothesis selection which are probabilistic and minimum description length-based.

2.5.1 Probabilistic hypothesis selection

This approach utilises ideas from probabilistic parsing of sentences or phrases (Jurafsky and Martin, 2000). It is assumed that rules and dictionary entries are independent of each other such that their probabilities can be multiplied. These probabilities are estimated from the training set by a maximum likelihood estimator. In general, if α denotes the left-hand side and β the right-hand side of a rule the probability is denoted as $Pr(\alpha \rightarrow \beta|\alpha)$. A parse t of a given word consists of a sequence of morpheme categories $c_1, \ldots, c_n$, with n being the number of categories, and each category c_i maps to an actual morpheme m_i the probability of the parse $Pr(t)$ is then defined as

$$Pr(w \rightarrow c_1, \ldots, c_n|w) \prod_{i=1}^{n} Pr(c_i \rightarrow m_i|c_i) \quad (1)$$

where w is the left-hand side of the most general rule. After having assigned a probability to each parse returned for a given word, we rank all parses by their probability. We do not limit ourselves to a single best parse and return the top k parses instead. We assume a certain degree of ambiguity which cannot be resolved for a list of words without syntactic context information.

2.5.2 MDL-inspired hypothesis selection

Our second approach for hypothesis selection builds on the *minimum description length (MDL)* principle. In general, MDL is applied in model selection for a trade-off between goodness-of-fit and complexity of the model given the data (Grunwald et al., 2005). We apply this principle heuristically by ranking parses using the frequency of their abduced morphemes. By selecting the top k parses we find new morphemes which occur in as many other word analyses as possible. In this way, we minimise the size of the updated dictionary. The advantage of the MDL-inspired approach over the probabilistic hypothesis selection is that the former selects parses based on a global evaluation where the latter solely depends on the distribution of rules and morphemes in the training set which need to be representative.

2.6 Post-processing

The post-processing step is concerned with merging morphological analyses and the original word list. Words which could not be parsed abductively nor inductively are analysed as single morpheme words. Previously replaced characters are substituted by their original ones. The list of Finnish words and analyses is transformed into its initial encoding again ('latin 1'). Hyphenated words in the original English word list are mapped to combinations of their single analyses.

3 Results and evaluation

We have applied the above described algorithm DEAP on the languages English, Finnish and Turkish. In Table 1 we show for each language how many parses per word were generated, how

Language	Generated	Hypothesis	Selected	MC metric			EMMA metric		
	parses	selection	parses	Prec.	Rec.	F-m.	Prec.	Rec.	F-m.
English	12.81±13.81	Probabilistic, no categories	1.76±0.99	0.5519	0.6078	0.5785	0.5260	0.6577	0.5845
		Probabilistic, categories	4.71±2.02	0.5196	0.5336	0.5265	0.2107	0.6498	0.3182
		MDL, no categories	1.76±1.08	0.5317	0.8135	**0.6431**	0.5195	0.7939	**0.6280**
		MDL, categories	4.29±2.17	0.5439	0.7402	0.6271	0.2653	0.7758	0.3954
Finnish	28.28±18.21	Probabilistic, no categories	2.36±0.83	0.6821	0.5289	0.5958	0.5733	0.6381	0.6039
		Probabilistic, categories	5.34±1.59	0.6611	0.5117	0.5769	0.4972	0.6404	0.5598
		MDL, no categories	3.29±1.48	0.7098	0.6703	0.6895	0.5642	0.7215	**0.6332**
		MDL, categories	5.33±1.60	0.7683	0.6480	**0.7030**	0.5017	0.7262	0.5934
Turkish	11.09±7.94	Probabilistic, no categories	3.49±1.52	0.6913	0.5873	0.6350	0.3205	0.4881	0.3869
		Probabilistic, categories	4.68±1.76	0.7352	0.5782	0.6473	0.2112	0.5403	0.3037
		MDL, no categories	3.17±1.46	0.7463	0.5867	0.6570	0.4567	0.5576	**0.5021**
		MDL, categories	4.88±1.74	0.8339	0.5636	**0.6726**	0.2093	0.5673	0.3058

Table 1: Results of DEAP with probabilistic and MDL hypothesis selection.

many were ultimately selected and the result in terms of precision, recall and f-measure. Since the MC metric possesses certain shortcomings we also state the performance using the EMMA metric introduced by Spiegler and Monson (2010).

As one can see from the number of generated hypotheses, words can have, for instance in Finnish, up to 28.28 parses on average. This is caused by the lack of syntactic information which would limit the search space and by augmenting the grammar for abductive parsing. The more abducibles are allowed, the more parses will be generated. For our experiments, we limited abduction to at most 5 abducible morpheme categories.

The MDL hypothesis selection performed best across all three languages. In contrast to the probabilistic approach, it evaluates each new morpheme globally by its frequency across the entire word list which guarantees more sensible choices. The probabilistic selection, in contrast, entirely depends on the representativeness of the training set.

Our two hypothesis selection approaches accepted all deductively found parses where no morphemes had to by hypothesised. If no parses could be found deductively the top k abductively found parses would be chosen. For k we used the value 6 which was determined by a cost-benefit analysis on the development set. In case that neither deductive nor abductive reasoning would find a parse, the algorithm would return the word as a morphemic singleton.

We also report results with and without morpheme categories since we believe that an evaluation should consider both perspectives separately. The algorithmic performance regarding word decomposition can be distorted by the algorithm's ability to label morphemes.

4 Related work

There are a number of approaches using inductive logic programming (ILP). ILP is concerned with learning rule-based concepts from examples and background knowledge. Representatives are FOIDL (Mooney and Califf, 1995) and CLOG (Manandhar et al., 1998) as ILP systems for learning decision lists. There also exist general systems for abductive (Ray and Kakas, 2006) and hybrid abductive-inductive inference (Ray, 2005). The former finds explanations for examples and an incomplete background knowledge and the latter also inductively generalises the abductive explanations. The algorithm described in this paper adds two automatic hypothesis selection techniques to the abductive method described in (Spiegler et al., 2010).

5 Conclusions

We have presented the algorithm DEAP which performs deductive-abductive parsing, generates a set of morphological hypotheses and selects the top k ones using either a probabilistic or an minimum description-based approach. We report the best results in terms of f-measure for English with 0.6431, Finnish with 0.7030 and Turkish with 0.6726 based on the MC 2010 metric.

Acknowledgments

The work was sponsored by EPSRC grant EP/E010857/1 *Learning the morphology of complex synthetic languages* and the Exabyte Informatics research theme using the BlueCrystal HPC facilities of the Advanced Computing Research Centre, University of Bristol – http://www.bris.ac.uk/acrc/.

References

Walter Daelemans and Antal van den Bosch. 2005. *Memory-Based Language Processing (Studies in Natural Language Processing)*. Cambridge University Press, New York, NY, USA.

P.A. Flach and A.C. Kakas, 2000. *Abduction and Induction*, chapter Abductive and inductive reasoning: background and issues, pages 1–27. Kulwer Academic Publishers.

Peter Flach. 1994. *Simply Logical*. John Wiley.

Peter D. Grunwald, In Jae Myung, and Mark A. Pitt, editors. 2005. *Advances in Minimum Description Length: Theory and Applications*. The MIT Press.

Daniel Jurafsky and James H. Martin. 2000. *Speech and Language Processing: An introduction to natural language processing, computational linguistics, and speech recognition.* Prentice Hall.

Suresh Manandhar, Saso Dzeroski, and Tomaz Erjavec. 1998. Learning multilingual morphology with CLOG. *Proc. of the 8th International Workshop on Inductive Logic Programming (ILP-98).*

R.J. Mooney and M.E. Califf. 1995. Induction of first-order decision lists: Results on learning the past tense of english verbs. *Proceedings of the 5th International Workshop on Inductive Logic Programming*, pages 145–146.

Charles S. Peirce. 1958. *Collected papers of Charles Sanders Pierce*. Harvard University Press.

Oliver Ray and Antonis Kakas. 2006. Prologica: a practical system for abductive logic programming. *Proceedings of the 11th International Workshop on Non-monotonic Reasoning*.

Oliver Ray. 2005. *Hybrid Abductive Inductive Learning*. Ph.D. thesis, Department of Computing, Imperial College London, December.

Sebastian Spiegler and Christian Monson. 2010. EMMA: A Novel Metric for Morphological Analysis. *Proceedings of the 23rd International Conference on Computational Linguistics (COLING).*

Sebastian Spiegler, Andrew van der Spuy, and Peter A. Flach. 2010. Ukwabelana - an open-source morphological Zulu corpus. *Proceedings of the 23rd International Conference on Computational Linguistics (COLING).*

Word decomposition with the Promodes algorithm family bootstrapped on a small labelled dataset

Sebastian Spiegler, Bruno Golénia, Peter A. Flach
Intelligent Systems Group
University of Bristol, UK
{spiegler, goleniab, flach}@cs.bris.ac.uk

Abstract

For the Morpho Challenge 2010 we present an algorithm family consisting of *Promodes*, *Promodes-H* and the ensemble *Promodes-E* which we tested on the languages English, Finnish and Turkish. The algorithms are based on a probabilistic generative model whose parameters are estimated from a small labelled dataset using maximum likelihood estimates. The algorithms are subsequently calibrated and applied to a large word list.

1 Introduction

The Morpho Challenge (MC) is concerned with morphological analysis in terms of decomposing words into their morphemes and labelling morphemes by their categories, if possible. In contrast to previous challenges, MC 2010 provided a small subset of 1000 decomposed and morphologically labelled words for each of the languages English, Finnish and Turkish. This data could be used to train an algorithm for morphological analysis either in a semi- or in a supervised manner.

Our participating algorithms called *Promodes*, *Promodes-H* and *Promodes-E* used the above described training sets to estimate their model parameters and were calibrated on a separate labelled development set.

2 Background

The *Promodes* algorithm family follows a line of algorithms for morphological analysis. Goldsmith (2001) presented *Linguistica* which attains signatures from an unlabelled word list without further information and applies them to predict a word's morpheme structure. Monson (2008) built a similar algorithm, called *Paramor*, which learns paradigmatic structures as sets of mutually substitutable morphological operations. Creutz (2006) developed the unsupervised algorithm family *Morfessor* which applies either minimum description length (MDL) or a probabilistic maximum a posteriori (MAP) framework.

More recent approaches are the following. *MorphoNet* (Bernhard, 2009) discovers transformation rules and builds a lexical network in order to cluster related words. Can and Manandhar (2009) induced paradigms from part-of-speech tagged words. Lavallee and Langlais (2009) performed morphological analysis based on formal analogies. *Ungrade* (Golenia et al., 2009) decomposes words by finding a pseudo-stem and analysing pre- and suffix sequences in a morpheme graph. The cognitive approach by Lignos et al. (2009) revisits the base-and-transform model described by Chan (2008). And *Metamorph* (Tchoukalov, 2009) is an analysis algorithm based on multiple sequence alignment.

The remainder of the paper is structured as follows. In Section 3 we give a brief introduction to the *Promodes* algorithm family. More detailed descriptions can be found in (Spiegler et al., 2010) and (Spiegler and Flach, 2010). In Section 4 we will state and interpret results on the languages English, Finnish and Turkish. In Section 5 we will draw our conclusions.

3 Promodes

The core of the *Promodes* family is a probabilistic generative model. The intuition of the model is that it describes the process of word generation from the left to the right by alternately using two dice. The first die decides whether a morpheme boundary is placed in the current word position whereas the second one identifies the corresponding letter transition. If all parameters of the model are known this process can be reversed to decode the underlying sequence of tosses which determine the morphological structure of a word.

Model parameters can be estimated in an unsupervised fashion using the expectation maximisation algorithm (Dempster et al., 1979). This approach was put to test in the MC 2009 and described in (Spiegler et al., 2010). Parameters can also be obtained from a small labelled dataset and then applied to a large unlabelled list of words by reverting to maximum likelihood estimates. In (Spiegler and Flach, 2010) we demonstrated the performance of different *Promodes* versions on the South African language Zulu in this setup.

The original *Promodes* algorithm uses a zero-order model for morpheme boundaries and a first-order model for letter transitions. *Promodes-H* represents a further development with an increased memory. Morpheme boundaries are modelled in first-order fashion – a boundary in a certain position also depends on the previous position. Moreover, letter transitions were extended to consider the previous letter transition as well as the previous boundary or non-boundary (second-order). For Zulu, we demonstrated that *Promodes-H* outperforms *Promodes* in the default setting.

We also introduced *Promodes-E* as an ensemble of the two algorithms above. In general, an ensemble consists of individually trained classifiers whose predictions are combined for new instances (Opitz and Maclin, 1999). Initially, *Promodes-E* averaged the probabilistic evidence given by *Promodes* and *Promodes-H* to arrive at a decision.

In (Spiegler and Flach, 2010) we subsequently performed experiments where we calibrated the decision threshold of the three algorithms. This was illustrated by precision-recall curves and hyperboles for constant f-measure values (isometrics). The highest f-measure isometric, tangent to the precision-recall curve, depicts the optimal decision threshold.

For the MC 2010, we have deployed calibrated *Promodes*, *Promodes-H* and their ensemble *Promodes-E* on the languages English, Finnish and Turkish.

4 Results

After having estimated model parameters of *Promodes* and *Promodes-H* on the labelled training set, we calibrated the two algorithms and their ensemble on the development sets of English, Finnish and Turkish. In Table 1 we give the numeric results for precision, recall and f-measure using the MC evaluation metric on the development set. We also state results using the EMMA metric since it was shown in (Spiegler and Monson, 2010) that the original MC metric lacks of robustness under certain circumstances.

Based on the MC metric, it can be seen that *Promodes* performs best and *Promodes-E* second best on all three languages. The best f-measure results are 0.5312 for English, 0.5121 for Turkish and 0.4412 for Finnish. Following the EMMA metric, the ranking of the three algorithms only changes for English where *Promodes-E* outperforms *Promodes*. Nevertheless, for English and Finnish, f-measure results increase by 0.0985 and 0.0344 on average whereas for Turkish the overall results drop by 0.0684 on average. We believe that the evaluation results for English and Finnish improve with EMMA due to its hard morpheme assignment and the fact that both languages have 1.10 ± 0.35 and 1.21 ± 0.52 analyses per word in the gold standard respectively. Each *Promodes* algorithm is rewarded for predicting a single analyses per word. Turkish, in contrast, has 2.07 ± 1.53 analyses per word in the gold standard and our algorithms are punished for returning only one.

In Figure 1 we illustrate the algorithms' performances on each language across all decision thresholds using the MC metric. We can see that *Promodes-H* performs worst across all languages, especially on English. This suggests that fewer morphological processes are captured with our higher order model – in contrast to results of the lower order model and oppositely to findings on Zulu in (Spiegler and Flach, 2010).

Furthermore, we can see that on English and Turkish *Promodes-E* follows *Promodes* relatively closely, however, there is not much difference in performance among all three algorithms on Finnish.

5 Conclusions

For the Morpho Challenge 2010, we have performed experiments with calibrated *Promodes*, *Promodes-H* and *Promodes-E* on the languages English, Finnish and Turkish.

In summary, although the algorithms of the *Promodes* family have been developed language-independently with as little structural and linguistic assumptions as possible their performance differs across languages considerably. In (Spiegler and Flach, 2010), the calibrated *Promodes* and *Promodes-H* achieved similar results on Zulu but both were outperformed by their calibrated ensem-

Language	Algorithm	Decision	MC metric			EMMA metric		
		threshold	Precision	Recall	F-measure	Precision	Recall	F-measure
English	Promodes	0.32	0.4313	0.6912	**0.5312**	0.5363	0.7021	0.6081
	Promodes-H	0.26	0.3006	0.6683	0.4147	0.4150	0.6417	0.5040
	Promodes-E	0.32	0.4830	0.4843	0.4837	0.5947	0.6326	**0.6131**
Finnish	Promodes	0.35	0.4034	0.4869	**0.4412**	0.4270	0.5480	**0.4800**
	Promodes-H	0.31	0.3421	0.5159	0.4114	0.3676	0.5233	0.4318
	Promodes-E	0.34	0.4197	0.4429	0.4310	0.4333	0.5254	0.4749
Turkish	Promodes	0.35	0.4853	0.5419	**0.5121**	0.4592	0.4003	**0.4277**
	Promodes-H	0.22	0.4498	0.3990	0.4229	0.4216	0.3446	0.3792
	Promodes-E	0.25	0.4186	0.5563	0.4777	0.4139	0.3881	0.4006

Table 1: Results for different language-algorithm combinations.

ble. On the datasets provided by the Morpho Challenge 2010, on the contrary, the calibrated lower-order version outperformed the higher-order and the ensemble algorithm across all three languages. This suggests that language characteristics which the *Promodes* algorithm family is based on differ across English, Finnish, Turkish in comparison to Zulu.

Acknowledgments

The work was sponsored by EPSRC grant EP/E010857/1 *Learning the morphology of complex synthetic languages* and the Exabyte Informatics research theme using the BlueCrystal HPC facilities of the Advanced Computing Research Centre, University of Bristol – `http://www.bris.ac.uk/acrc/`.

References

Delphine Bernhard. 2009. Morphonet: Exploring the use of community structure for unsupervised morpheme analysis. *Working notes for the CLEF 2009 Workshop, Corfu, Greece*.

Burcu Can and Suresh Manandhar. 2009. Unsupervised learning of morphology by using syntactic categories. *Working notes for the CLEF 2009 Workshop, Corfu, Greece*.

Erwin Chan. 2008. *Structures and Distributions in Morphology Learning*. Ph.D. thesis, Computer and Information Science, University of Pennsylvania.

Mathias Creutz. 2006. *Induction of the Morphology of Natural Language: Unsupervised Morpheme Segmentation with Application to Automatic Speech Recognition*. Ph.D. thesis, Helsinki University of Technology, Espoo, Finland.

Arthur Dempster, Nan Laird, and Donald Rubin. 1979. Maximum likelihood from incomplete data via the em algorithms. *Journal of the Royal Statistical Society*, 39:1–38.

John Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27:153–198.

Bruno Golenia, Sebastian Spiegler, and Peter Flach. 2009. Ungrade:unsupervised graph decomposition. *Working notes for the CLEF 2009 Workshop, Corfu, Greece*.

Jean-Francois Lavallee and Philippe Langlais. 2009. Morphological acquisition by formal analogy. *Working notes for the CLEF 2009 Workshop, Corfu, Greece*.

Constantine Lignos, Erwin Chan, Mitchell P. Marcus, and Charles Yang. 2009. A rule-based unsupervised morphology learning framework. *Working notes for the CLEF 2009 Workshop, Corfu, Greece*.

Christian Monson. 2008. *ParaMor: From Paradigm Structure To Natural Language Morphology Induction*. Ph.D. thesis, Language Technologies Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA.

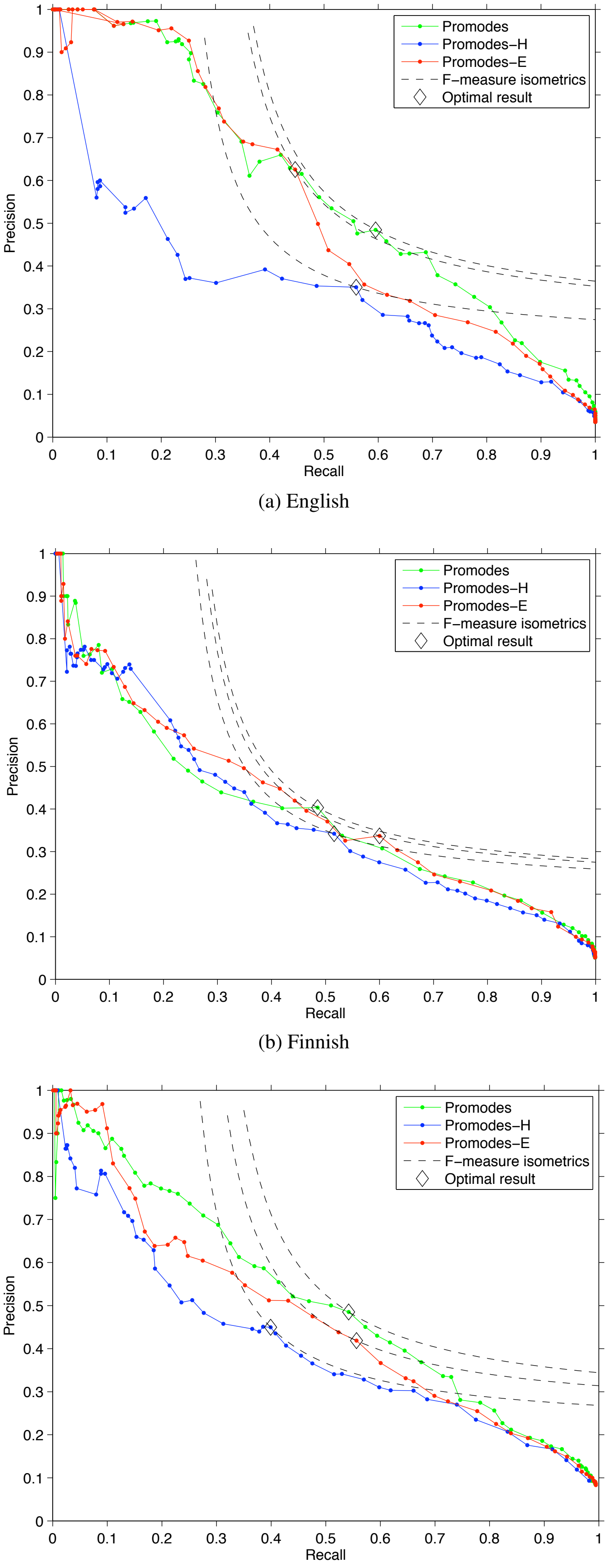
David Opitz and Richard Maclin. 1999. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198.

Sebastian Spiegler and Peter Flach. 2010. Enhanced word decomposition by calibrating the decision threshold of probabilistic models and using a model ensemble. *48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*.

Sebastian Spiegler and Christian Monson. 2010. EMMA: A Novel Metric for Morphological Analysis. *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*.

Sebastian Spiegler, Bruno Golenia, and Peter Flach. 2010. Unsupervised word decomposition with the promodes algorithm. *In Multilingual Information Access Evaluation Vol. I, CLEF 2009, Corfu, Greece, Lecture Notes in Computer Science, Springer*.

Tzvetan Tchoukalov. 2009. Multiple sequence alignment for morphology induction. *Working notes for the CLEF 2009 Workshop, Corfu, Greece*.



(a) English

(b) Finnish

(c) Turkish

Figure 1: Precision-recall curves for different language-algorithm combinations on the development set using the MC metric.

TKK REPORTS IN INFORMATION AND COMPUTER SCIENCE

TKK-ICS-R27 Antti Ajanki, Mark Billinghurst, Melih Kandemir, Samuel Kaski, Markus Koskela, Mikko Kurimo, Jorma Laaksonen, Kai Puolamäki, Timo Tossavainen
Ubiquitous Contextual Information Access with Proactive Retrieval and Augmentation. December 2009.

TKK-ICS-R28 Juho Frits
Model Checking Embedded Control Software. March 2010.

TKK-ICS-R29 Miki Sirola, Jaakko Talonen, Jukka Parviainen, Golan Lampi
Decision Support with Data-Analysis Methods in a Nuclear Power Plant. March 2010.

TKK-ICS-R30 Teuvo Kohonen
Contextually Self-Organized Maps of Chinese Words. April 2010.

TKK-ICS-R31 Jefrey Lijffijt, Panagiotis Papapetrou, Niko Vuokko, Kai Puolamäki
The smallest set of constraints that explains the data: a randomization approach. May 2010.

TKK-ICS-R32 Tero Laitinen
Extending SAT Solver With Parity Constraints. June 2010.

TKK-ICS-R33 Antti Sorjamaa, Amaury Lendasse
Fast Missing Value Imputation using Ensemble of SOMs. June 2010.

TKK-ICS-R34 Yoan Miche, Patrick Bas, Amaury Lendasse
Using Multiple Re-embeddings for Quantitative Steganalysis and Image Reliability Estimation. June 2010.

TKK-ICS-R35 Teuvo Kohonen
Contextually Self-Organized Maps of Chinese Words, Part II. August 2010.

TKK-ICS-R36 Antti Ukkonen
Approximate Top-k Retrieval from Hidden Relations. August 2010.

ISBN 978-952-60-3330-3 (Online)
ISSN 1797-5042 (Online)