Morphological Analysis by Multiple Sequence Alignment

Tzvetan Tchoukalov[1], Christian Monson[2], and Brian Roark[2]

[1] Stanford University
[2] Center for Spoken Language Understanding, Oregon Health & Science University
eurus@stanford.edu, monsonc@csee.ogi.edu, roark@cslu.ogi.edu

Abstract. In biological sequence processing, Multiple Sequence Alignment (MSA) techniques capture information about long-distance dependencies and the three-dimensional structure of protein and nucleotide sequences without resorting to polynomial complexity context-free models. But MSA techniques have rarely been used in natural language (NL) processing, and never for NL morphology induction. Our MetaMorph algorithm is a first attempt at leveraging MSA techniques to induce NL morphology in an unsupervised fashion. Given a text corpus in any language, MetaMorph sequentially aligns words of the corpus to form an MSA and then segments the MSA to produce morphological analyses. Over corpora that contain millions of unique word types, MetaMorph identifies morphemes at an F_1 below state-of-the-art performance. But when restricted to smaller sets of orthographically related words, MetaMorph outperforms the state-of-the-art ParaMor-Morfessor Union morphology induction system. Tested on 5,000 orthographically similar Hungarian word types, MetaMorph reaches 54.1% and ParaMor-Morfessor just 41.9%. Hence, we conclude that MSA is a promising algorithm for unsupervised morphology induction. Future research directions are discussed.

1 Introduction

Biologists are interested in the function of genes and proteins. Since organisms evolve by the slow mutation of individual base pairs in their DNA, gene regions from related organisms that consist of similar sequences of nucleotides will likely perform similar functions. Multiple Sequence Alignment (MSA) techniques are one suite of tools that computational biologists use to discover nucleotide sequences that are unusually similar, and that thus likely serve similar biological functions [1].

Like biologists, linguists are interested in the sub-regions of longer linear sequences that serve particular functions. Where biologists look at strings of nucleotide bases in DNA or sequences of amino acids in proteins, linguists examine the strings of phonemes or written characters that form words. And where biologists seek genes, linguistics identify morphemes—the smallest linguistic units that carry meaning. Given the similarities between biological and linguistic sequences, we seek to transfer the successes of MSA models from biology to induce natural language morphology in an unsupervised fashion.

C. Peters et al. (Eds.): CLEF 2009 Workshop, Part I, LNCS 6241, pp. 666–673, 2010.

Although we are inspired by biology, building MSAs for natural language morphology induction is fundamentally different from building MSAs in biological applications. In biology, it is typical to align a few sequences (on the order of 10) of very long length (perhaps millions of base pairs). In our NL morphology application, the sequences are words and are thus relatively short (on the order of 10 characters)—but there may be tens of thousands or even millions of distinct word types to align. Moreover, our goals are somewhat different from the goals that biologists typically have when applying MSA techniques. We wish to definitively segment words into separate morphemes, but we have not encountered any work in computational biology that uses MSA to segment out genes. Instead, biologists use MSA to merely identify regions of likely similarity between sequences.

Relating our MSA work to research directions in NL morphological processing: While we are unaware of any prior attempt to model the structure of NL morphology using MSA techniques, MSA is at base a method for measuring distances between strings—and string edit distances *have* played a part in a variety of unsupervised morphology induction systems. Baroni et al. [2], for example, seed a semantically based induction system with pairs of words that are orthographically similar. Likewise, Wicentowski [3] trains a statistical model of morphological structure from several weak correlates of morphological relatedness—including the Levenshtein distance between pairs of words. Readers interested in unsupervised morphology induction more broadly may consult Chapter 2 of [4].

2 The MetaMorph Multiple Sequence Alignment Algorithm

The input to an MSA algorithm is a set of sequences; and the output is an alignment of the elements of the sequences. Fig. 1 depicts an alignment over a set of ten English words. Each sequence, i.e. each word, in Fig. 1 forms a separate row of the alignment table. An MSA algorithm places the characters of each sequence into aligned columns. The order of elements in each sequence is fixed, but, to improve an alignment, an MSA algorithm may place gaps, '-', in some columns between the characters of a sequence. The ten sequences of Fig. 1 are arranged into eight aligned columns.

A variety of algorithms could produce a multiple sequence alignment like that in Fig. 1. Our MetaMorph algorithm employs two standard MSA algorithms in turn: *Progressive Alignment* [5] and a *Profile Hidden Markov Model (HMM)* [1]. Both Progressive Alignment and Profile HMMs define position specific distributions over characters. Fig. 2 displays the position specific character distributions for the alignment in Fig. 1. For each of the eight columns of the alignment table, Fig. 2 has a corresponding column that contains a smoothed probability distribution over the alphabet of characters that appears in Fig. 1: Each column distribution in Fig. 2 contains a count for each occurrence of each character in the corresponding column of Fig. 1, plus a Laplace smoothing constant of one for each character. We treat the gap as simply another character of the alphabet. The probability of a character given a column is the ratio of the character count in that column's distribution to the distribution total. For example, in Figs. 1 and 2, the probability of the character 'd' given column 1 is 5/26.

```
12345678
d-anc-es
d-anc-ed
d-anc-e-
d-ancing
r-unning
j-umping
j-ump-ed
j-ump-s-
j-ump---
laughing
```

Fig. 1. A sample multiple sequence alignment (MSA)

Chars	1	2	3	4	5	6	7	8
a	1	2	5	1	1	1	5	1
c	1	1	1	1	5	1	1	1
d	5	1	1	1	1	1	1	3
e	1	1	1	1	1	1	1	1
g	1	1	1	2	1	1	1	5
h	1	1	1	1	2	1	1	1
i	1	1	1	1	1	5	1	1
j	5	1	1	1	1	1	1	1
l	2	1	1	1	1	1	1	1
m	1	1	1	5	1	1	1	1
n	1	1	1	6	2	1	5	1
p	1	1	1	1	5	1	1	1
r	2	1	1	1	1	1	1	1
s	1	1	1	1	1	1	2	2
u	1	1	7	1	1	1	1	1
gap	1	10	1	1	1	7	2	4

Fig. 2. Laplace-smoothed (count plus 1) position specific character distributions for the MSA in Fig. 1

2.1 Building an Initial MSA via Progressive Alignment

MetaMorph begins with a progressive alignment algorithm that builds an initial alignment over an orthographically similar subset of the full input corpus. Progressive alignment algorithms build an alignment for a set of sequences iteratively. After a first pair of sequences are aligned to each other, a third sequence is aligned to the newly formed alignment; a fourth sequence follows; and a fifth, etc. The size of the orthographically similar subset of corpus words is a free parameter. We experimented with subsets that contained between 5,000 and 20,000 words.

Step 1: Ordering. Before MetaMorph aligns words, our algorithm orders the words which will form the initial MSA. The first two words in the ordered list are the Levenshtein most similar pair of words from the 1000 most frequent words of the input corpus. MetaMorph then sequentially adds words to our ordered list by identifying, from all the words in the input corpus, the word that is most similar to some word already in the ordered list. To ensure that the initial MSA contains standard natural language words, we require all words in the ordered list to be between 5 and 16 characters in length and to contain no hyphens or numbers. MetaMorph continues to add words to the ordered list until a preset size limit is reached.

Step 2: Alignment. To produce an MSA from the ordered list of words, MetaMorph initializes an MSA to the first word in the sorted list. Each character in the first word appears in a separate column. Using Laplace smoothing, MetaMorph then calculates the (trivial) column distributions over the characters of the alphabet, a la Fig. 2.

For each remaining word, *w*, in the ordered list, MetaMorph uses a dynamic programming algorithm to, in turn, identify the lowest-cost alignment of *w* to the current MSA. Beginning with the first character of *w* and the first column of the MSA, there are three possible alignment choices: First, the character may be aligned to the column; Second, the column may be aligned to a gap that is inserted into *w*; and third, the character may be aligned to a column of gaps that is inserted into the MSA. We

define the cost of matching a character, *c*, to an MSA column, *l*, to be the negative logarithm of the probability of *c* occurring in *l*. Treating gaps as just another character, the cost of aligning the column *l*, to a gap in the word *w* is simply the negative logarithm of the probability of a gap in *l*. And we measure the cost of matching a character, *c*, to a newly-inserted column of gaps as the negative logarithm of the (smoothed) probability of *c* appearing in a column that thus far contains only gaps. As the number of words in the alignment increases, the contribution of Laplace smoothing to the overall character distribution decreases, and hence the cost of inserting a column of gaps into the MSA increases.

The score of an alignment of a word, *w*, to an MSA is the sum of the match costs and gap insertions costs specified by that particular alignment. Dynamic programming with back-tracing finds the optimal alignment of each *w* in O(NM) time, where N is the length of *w*, and M the length of the current MSA. When all words in the ordered wordlist have been inserted into the MSA, the initial alignment cycle is complete.

Step 3: Realignment. After the initial alignment phase, MetaMorph performs leave-one-out refinement [6]: Sequentially, MetaMorph removes each word from the MSA, and then realigns the removed word to the remaining MSA. MetaMorph halts leave-one-out refinement when one of two criteria is met: 1. The MSA remains unchanged, or 2. The sum of the entropies of the column distributions increases after a set number of realignment cycles. Leave-one-out refinement is designed to specialize each column on a smaller selection of characters. If entropy rises, then the columns are permitting a wider variety of characters, and we halt realignment.

2.2 Finalizing Alignment via a Profile HMM

Once MetaMorph's progressive alignment phase has built an alignment over an orthographically similar subset of the full corpus, our algorithm freezes the initial alignment as a Profile HMM. Each column of the progressively built alignment acts as an HMM state whose character production probabilities correspond to the column's character distribution. Each word of the corpus that was not in the original orthographically similar subset is then aligned to the Profile HMM.

2.3 Segmentation

Having obtained a final MSA, we must now produce a morphological segmentation of the input corpus. To motivate the segmentation strategy that the MetaMorph algorithm employs, Fig. 3 depicts six sequences of an MSA built over a Hungarian corpus of 500,000 words. The first five sequences in Fig. 3 are legitimate inflections of the word in the first sequence: 'között' is a postposition in Hungarian, while 'i', 't', 'e', and 'em' are suffixes that attach to postpositions. In contrast, the last sequence in Fig. 3, 'kötöttem' is an inflected verb that is linguistically unrelated to the other sequences. A reasonable segmentation of the final sequence would be 'köt-ött-em', with 'köt' a verb stem meaning 'tie', 'ött' marking past tense, and 'em' 1st person singular.

To segment an MSA into morphemes, our MetaMorph algorithm selects a set of columns in the MSA and segments *all* the words of the corpus at those columns. Examining Fig. 3, the pattern of gap columns does not appear to indicate where a morpheme boundary should be placed: columns of gaps separate *all* the characters of all

```
-----k----ö---z-----ö-------t-----------t---------
-----k----ö---z-----ö-------t-----------t----i----
-----k----ö---z-----ö-------t-----------t----i-t--
-----k----ö---z-----ö-------t-----------t----e----
-----k----ö---z-----ö-------t-----------t----e-m--
-----k----ö---t-----ö-------t-----------t----e-m--
```

Fig. 3. Six Sequences from an MSA induced over a Hungarian corpus

the words, with long sequences of gap columns sometimes occurring internal to a morpheme (eleven gaps separate the doubled 't's for example). Other obvious techniques, such as segmenting at columns with the maximal gap probability or at those columns whose probability distributions had minimal entropy, generated completely implausible segmentations.

Lacking a segmentation strategy that relies solely on the MSA, MetaMorph instead leverages knowledge from an independent algorithm for unsupervised induction of morphology. The independent system which we used is the ParaMor-Morfessor Union system from Morpho Challenge 2009 [7]. The ParaMor-Morfessor system placed first at F_1 for morpheme identification in three of the five languages of Morpho Challenge 2009. To segment the words of a corpus, the MetaMorph algorithm searches for a set of segmentation columns that maximize the F_1 score *against the independent (still unsupervised) system*.

MetaMorph uses a greedy search algorithm to decide upon a set of segmentation columns. One at a time, MetaMorph considers each column in the MSA as a potential segmentation point. The segmentations that result from a particular column are scored against the analyses provided by the independent morphology segmentation algorithm. MetaMorph retains that segmentation column which most improves F_1, and then iteratively considers adding a second segmentation column. If, after any iteration, no column is found to improve F_1, MetaMorph terminates the search for additional segmentation columns. Although the segmentation columns are fixed for all words, the final number of morphemes in any particular word will still vary because, after segmenting a word, MetaMorph discards morphemes that consist solely of gaps.

3 Results and Conclusions

To evaluate the success of the MetaMorph algorithm, we participated in Morpho Challenge 2009 [8], where ten groups from around the world assessed their unsupervised morphology induction systems with both linguistic and task-based criteria. The linguistic evaluation of Morpho Challenge measured the precision, recall, and F_1 score of each unsupervised algorithm at identifying the constituent morphemes of the words in a corpus. Of the six tracks of the linguistic competition, MetaMorph had the least success at the two Arabic scenarios and the most success on Turkish. MetaMorph's poor performance on Arabic, less than 6% F_1 for both the vowelized and unvowelized tracks, is directly attributable to MetaMorph's reliance on the ParaMor-Morfessor Union system for its segmentation strategy. The Union system also suffered its poorest performance on the Arabic tracks, F_1 scores of less than 10%.

In Turkish, MetaMorph outperformed at F_1 the baseline unsupervised system of Morpho Challenge, a system named Morfessor [9]: MetaMorph achieved an F_1 score of 33.6% where Morfessor came in at 29.7%. Backstopping MetaMorph's comparatively strong performance on Turkish is the highest absolute recall score that MetaMorph attained for any language, 29.5%. But since absolute scores of morpheme identification are not comparable across languages, consider MetaMorph's recall as a fraction of the recall score (60.4%) of that system [7] which had the highest F_1 score at Morpho Challenge for Turkish: this recall fraction is 0.488 (i.e. 29.5% / 60.4%). If we calculate MetaMorph's recall fraction against the F_1-best system for the other non-Arabic languages of Morpho Challenge we get 0.401 for English, but just 0.322 for German and 0.300 for Finnish, Interestingly, there are many fewer word types in the Turkish and English data sets (617,000 and 385,000 respectively) of Morpho Challenge than there are in the Finnish and German sets (2.21 and 1.27 million respectively). The following experiments suggest, counterintuitively, that it may be the smaller size of the Turkish and English data sets that lead to MetaMorph's higher recall.

We used Hunmorph [10], a hand-built Hungarian morphological analyzer, to produce a morphological answer key containing 500,000 unique Hungarian word types from the Hunglish corpus [11]. Over the full Hungarian corpus, MetaMorph's F_1 score reached a paltry 19.7%. But we found that if we restricted our evaluation to just those words from which MetaMorph's progressive alignment algorithm constructed the initial MSA, performance improved dramatically.

We ran three experiments. In the first experiment, MetaMorph's progressive alignment algorithm built an MSA from a set of 5,000 orthographically similar words, i.e., the size limit in step 1 of Section 2.1 is set to 5,000; in the second, MetaMorph used a size limit of 10,000 words; and in the third, a size limit of 20,000. During the segmentation phase of these three experiments, we instructed MetaMorph's greedy search to select segmentation columns that maximized F_1 score, *not* over the full Hungarian corpus, but rather over just the words in the smaller set of orthographically similar words. Using the Linguistic evaluation procedure from Morpho Challenge, we then evaluated MetaMorph's morphological analyses over these same three (smaller) sets of orthographically similar words. Additionally, we evaluated the performance of the ParaMor-Morfessor Union system over these same three sets of words. To evaluate the Union system we used the full Hungarian corpus to induce segmentations, but restricted our evaluations to the sets of 5,000, 10,000, and 20,000 words. The results of these experiments appear in Fig. 4.

Immediately striking from Fig. 4 is that MetaMorph significantly outscores the ParaMor-Morfessor Union over the 5,000 and 10,000 word sets. Remember that MetaMorph's segmentation phase seeks to emulate the segmentations that the ParaMor-Morfessor Union system produces. Over small datasets, MetaMorph has successfully generalized beyond the system that is used as a segmentation guide.

Furthermore, as the set size increases in Fig. 4, MetaMorph's F_1 steadily decreases. Since each of our three experiments is learning from and evaluating over a different set of words, one explanation for MetaMorph's downward trend might simply be that the set of 20,000 words contained more inherent morphological complexity than the smaller sets. But this explanation fails when we examine the performance of the ParaMor-Morfessor Union system over these same data sets: The Union system's

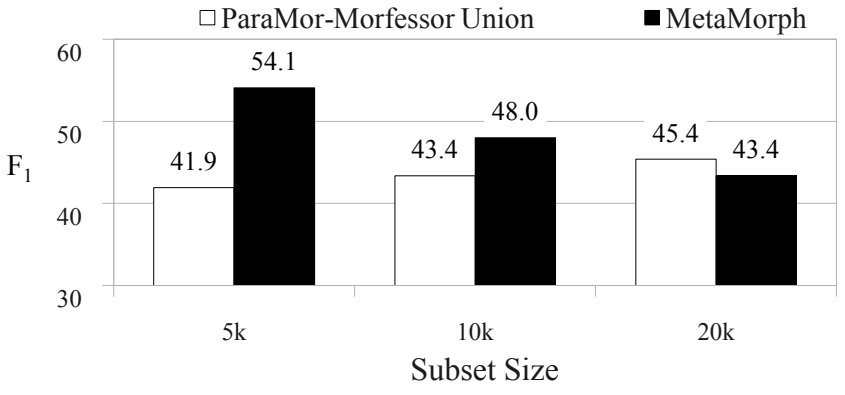

Fig. 4. The F_1 performance of two unsupervised morphology induction algorithms for three subsets of a Hungarian corpus

performance increases in step with the set size. Instead, MetaMorph's strong performance at the smallest word-set sizes, leads us to conclude that MSA is most effective when used over a smaller set of words that exhibit orthographic similarity. This same effect may be at work in MetaMorph's higher recall scores for English and Turkish in Morpho Challenge proper.

For some tasks, such as machine translation (MT), MetaMorph's conservative lower-recall approach to morphological segmentation pays off: MetaMorph took second place in Finnish MT at Morpho Challenge 2009, with a BLEU score of 28.20. In the MT evaluation the words of a non-English language text were replaced by their automatic morphological analyses before applying a statistical MT algorithm. The baseline statistical MT system that translates directly from words had a BLEU score of 27.64; thus MetaMorph improves BLEU score over the word-based model by 0.56 BLEU points. For exhaustive details about the absolute and relative performance of the MetaMorph algorithm at Morpho Challenge 2009, see [8].

The Next Steps. MetaMorph's success at analyzing the morphological structure of smaller, more focused, sets of words suggests that in future we use progressive alignment techniques to build a number of separate alignment structures focused on different subsets of the full corpus. Each subset of the corpus would contain orthographically similar words. It may also be the case that the optimal number of words for which to build an alignment is smaller than 5,000. Indeed, the optimal set size may vary by language, or even by part-of-speech within a language.

A second weakness that we would like to address in the MetaMorph algorithm is the poor correlation between the arrangement of columns in the MSA and the arrangement of morphemes within words. Where most morphemes are contiguous sequences of characters, MetaMorph's MSA columns place gap symbols internal to true morphemes. Better correlation between MSA columns and morphemes in words would free MetaMorph from relying on an independent unsupervised morphology induction system during segmentation (Section 2.3). A match cost function that accounted for string position is one potential method for producing contiguous morpheme columns. We might, for example, lower match costs (and thereby reduce the

number of gap columns) near the middle of an MSA to allow for the wide variety of characters that are likely in the stems of words. Another possible solution may be to tie the character distributions of neighboring MSA columns so as to avoid overtraining a column to a particular character.

Acknowledgements. This research was supported in part by NSF Grant #IIS-0811745 and DOD/NGIA grant #HM1582-08-1-0038. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the NSF or DOD.

References

1. Durbin, R., Eddy, S.R., Krogh, A., Mitchison, G.: Biological Sequence Analysis: Probabilis-tic Models of Proteins and Nucleic Acids. Cambridge University Press, Cambridge (1998)
2. Baroni, M., Matiasek, J., Trost, H.: Unsupervised Discovery of Morphologically Related Words Based on Orthographic and Semantic Similarity. In: ACL Special Interest Group in Computational Phonology in Cooperation with the ACL Special Interest Group in Natural Language Learning (SIGPHON/SIGNLL), Philadelphia, Pennsylvania, pp. 48–57 (2002)
3. Wicentowski, R.: Modelling and Learning Multilingual Inflectional Morphology in a Minimally Supervised Framework. Ph.D. Thesis, The Johns Hopkins University, Baltimore, Maryland (2002)
4. Feng, D.F., Doolittle, R.F.: Progressive Sequence Alignment as a Prerequisite to Correct Phylogenetic Trees. Journal of Molecular Evolution 25(4), 351–360 (1987)
5. Gotoh, O.: Significant Improvement in Accuracy of Multiple Protein Sequence Alignments by Iterative Refinement as Assessed by Reference to Structural Alignments. Journal of Molecular Biology 264(4), 823–838 (1996)
6. Monson, C., Hollingshead, K., Roark, B.: Simulating Morphological Analyzers with Stochastic Taggers for Confidence Estimation. In: 10th Workshop of the Cross-Language Eval-uation Forum, CLEF 2009, Corfu, Greece, Revised Selected Papers. LNCS, Springer, Heidelberg (2010)
7. Kurimo, M., Virpioja, S., Turunen, V.T., Blackwood, G.W., Byrne, W.: Overview and Results of Morpho Challenge 2009. In: 10th Workshop of the Cross-Language Evaluation Forum, CLEF 2009, Corfu, Greece, Revised Selected Papers. LNCS, Springer, Heidelberg (2010)
8. Creutz, M.: Induction of the Morphology of Natural Language: Unsupervised Morpheme Segmentation with Application to Automatic Speech Recognition. Ph.D. Thesis, Computer and Information Science, Report D13, Helsinki, University of Technology, Espoo, Finland (2006)
9. Trón, V., Gyepesi, G., Halácsy, P., Kornai, A., Németh, L., Varga, D.: Hunmorph: Open Source Word Analysis. In: ACL Workshop on Software (2005)
10. Varga, D., Halácsy, P., Kornai, A., Németh, L., Trón, V., Váradi, T., Sass, B., Bottyán, G., Héja, E., Gyarmati, Á., Mészáros, Á., Labundy, D.: Hunglish Corpus (2009), `http://mokk.bme.hu/resources/hunglishcorpus` (accessed on August 18, 2009)