Unsupervised Morphological Analysis Using Tries

Koray Ak and Olcay Taner Yildiz

Abstract This article presents an unsupervised morphological analysis algorithm to segment words into roots and affixes. The algorithm relies on word occurrences in a given dataset. Target languages are English, Finnish, and Turkish, but the algorithm can be used to segment any word from any language given the wordlists acquired from a corpus consisting of words and word occurrences. In each iteration, the algorithm divides words with respect to occurrences and constructs a new trie for the remaining affixes. Preliminary experimental results on three languages show that our novel algorithm performs better than most of the previous algorithms.

Keywords Unsupervised learning · Morphology · Word decomposition

1 Introduction

Natural Language Processing (NLP) is a field of computer science that investigates interactions between computers and human languages. NLP is used for both generating human readable information from computer systems and converting human language into more formal structures that a computer can understand. Well known problems of NLP are morphological analysis, part of speech tagging, wordsense disambiguation, and machine translation.

K. Ak (✉) · O. T. Yildiz
Department of Computer Science and Engineering, Işik University,
34980 Şile, Istanbul, Turkey
e-mail: koray@isikun.edu.tr; korayak@gmail.com

O. T. Yildiz
e-mail: olcaytaner@isikun.edu.tr

E. Gelenbe et al. (eds.), *Computer and Information Sciences II*,
DOI: 10.1007/978-1-4471-2155-8_8,

Morphological analysis or decomposition studies the structure of the words and identifies the morphemes (smallest meaning-bearing elements) of the language. Any word form can be expressed as a combination of morphemes. For instance, the English word "enumeration" can be decomposed as e+number+ate+ion and "interchangeable" as inter+change+able, and the Turkish word "isteyenlerle" as iste+yen+ler+le. Generally words are known as the basic units of the language but morphemes are the smallest syntactic unit and they reveal the relationship between word forms. In this respect, morphological analysis investigates the structure, formation and function of words, and attempts to formulate rules that model the language.

Morphological analysis is widely used in different areas such as speech recognition, machine translation, information retrieval, text understanding, and statistical language modeling. In many languages this task is both difficult and necessary, due to the large number of different word forms found in the text corpus. Highly inflecting languages may have thousands of different word forms of the same root, which makes the construction of an affixed lexicon hardly feasible. As an alternative to the hand-made systems, there exist algorithms that work unsupervised manner and autonomously do morphological analysis for the words in an unannotated text corpus.

In this paper, an unsupervised learning algorithm is proposed to extract information about the text corpus and the model of the language. The proposed algorithm constructs a trie that consists of characters and the occurrences of the words as nodes. The algorithm then detects roots of the given words by examining the occurrences in the path of the word. When the root is revealed, the algorithm creates a new trie from the affix parts, left after the root for each word. The algorithm continues recursively until there is no affix left to process.

The paper is organized as follows: In Sect. 2, we present previous work in the field. In Sect. 3, we present proposed algorithm. We give the results of our experiments in Sect. 4 and conclude in Sect. 5.

2 Related Work on Unsupervised Morphological Analysis

Morpho Challenge [1] is one of the competitions of the EU Network of Excellence PASCAL2 Challenge Program working on unsupervised morphological disambiguation since 2005. The objective of the challenge is to design an unsupervised machine learning algorithm that discovers which morphemes the words consist of.

In Morpho Challenge 2005, Bernhard [2] propose a method that relies on transitional probabilities of each substring of the word in the lexicon, and distinguishes stems and affixes by examining the differences in lengths and frequencies of the words.

Keshava [3] used a simple approach to gather morphemes based on finding substring words and transitional probabilities. The algorithm constructs two trees;

a forward tree where each node from top to the leaf corresponds to a word in the corpus and a backward tree to find suffix probabilities easily.

For the Turkish task in 2007, Zeman [4] proposed a paradigm based approach. All possible suffix-stem pairs are grouped into paradigms. Since all possible segmentation points are considered, the number of paradigms is huge and they are filtered. In the segmentation phase; each possible segmentation of the word is searched in the paradigms.

ParaMor [5] dominated the Morpho Challenge 2008 in all languages. Each word is examined by segmenting from every character boundary. When two or more words end in the same word-final string, ParaMor constructs a paradigm seed. Paradigms are then expanded to full candidate paradigms by adding additional suffixes.

In 2009, Monson et al. [6] proposed an improved version of ParaMor [5]. In the original version, ParaMor did not assign a numeric score to its segmentation decisions. A natural language tagger is trained to score each character boundary in each word. Using ParaMor as a source of labeled data, finite-stage tagger is trained to identify, for each character, c, in a given word, whether or not ParaMor will place a morpheme boundary immediately before c.

3 REC-TRIE

The input of the Morpho Challenge is a corpus and word list of the words with frequencies as appeared in the corpus. Since character encodings differ in the datasets, some modifications are done. English dataset consists of standard text and all words are lower-cased. Finnish dataset uses ISO Latin 1 (ISO 8859-1). The Scandinavian special letters å, ä, ö are rendered as one-byte characters. Turkish dataset is standard text and all words except the letters specific to Turkish are lower-cased. The letters specific to the Turkish language are replaced by capital letters of the standard Latin alphabet, “açıkgörüşlülügünü” is converted to “aCIkgOrUSlUlUGUnU”.

As mentioned in the first section, one of the problems in unsupervised morphological analysis is the data sparsity. Given a large dataset, most of the root words appear in the corpus. For example, in the datasets of challenge, there exist 15545 root words among 617298 words where total root count for Turkish is 23470 [7], that is 66% of the roots appeared in the dataset.

The pseudo code of our proposed algorithm (REC-TRIE) is given in Fig. 1. We simply populate word trie with words from the list that occurred more than 5 times and store the corresponding character, the number of occurrence in the corpus and the number of times that character is used in this path in this depth (Line 3). With the fact explained above, we assume the smallest most occurred word in a path is the root of the words in the path.

Once the algorithm finds root morphemes in each word, it saves the corresponding segmentation into a table and continues to the next iteration (Line 7).

```
1  Read words from Wordlist W
2  i = 1
3  Construct word Trie_i
4  Construct word Table
5  Do until no unsegmented words remain
6      For each word in the Table
7          Find boundary for unsegmented part with Trie and update Table
8          If the word is not fully segmented
9              Add unsegmented part to Trie_{i+1}
10         End If
11     End For
12 i = i + 1
13 End Do
```

Fig. 1 The pseudocode of REC-TRIE

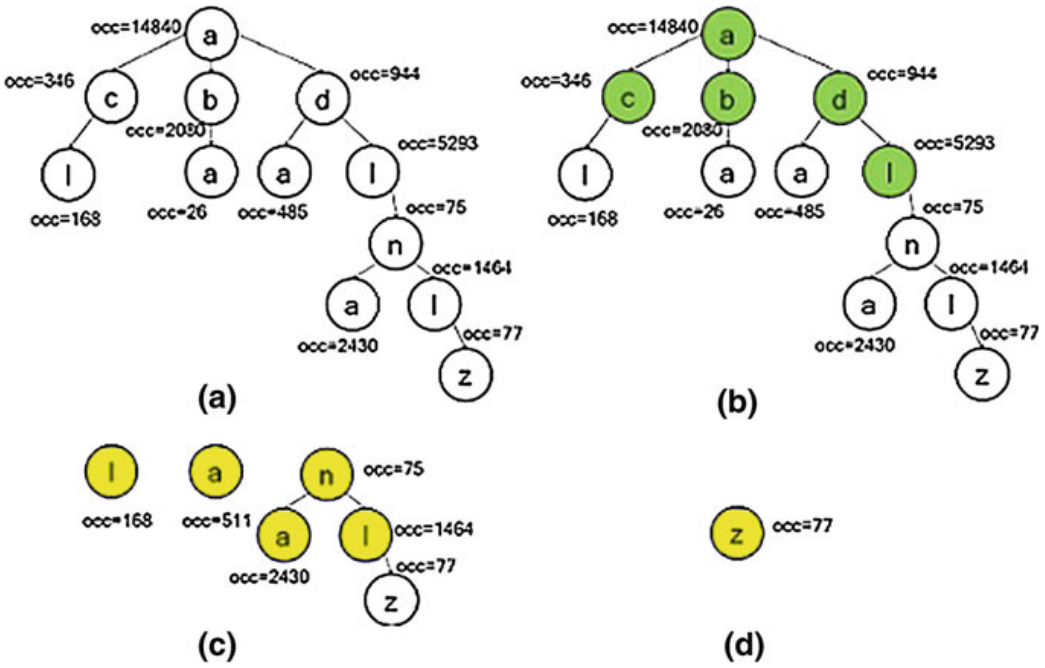

Fig. 2 Sample run from REC-TRIE. **a** A sample trie initialized. **b** After first iteration root words detected. **c** First affix trie is constructed with the extracted affix parts left from roots and first affixes are found. **d** Second iteration REC-TRIE created new affix tree and found the last affix

In each iteration, the rest part of the word is put on a new trie (Line 9) and affix boundaries are found recursively with the same method applied for root extraction. The algorithm continues until no affix candidate is left.

We present a sample run of REC-TRIE in Fig. 2. After initializing word trie the algorithm traverses each path and chooses the most occurred smallest word as root. The word "ada" is segmented as ad + a since the most occurred character is d with 944 occurrence in the path. However the words "adI", "adIn", "adIna", "adInI",

Table 1 Precision, Recall, and F-Measure of REC-TRIE compared with other algorithms in Morpho Challenge 2009 for Turkish

Author	Method	Precision (%)	Recall (%)	F-Measure (%)
Monson et al.	Paramor-Morfessor Mimic	48.07	60.39	53.53
Monson et al.	ParaMor-Morfessor Union	47.25	60.01	52.88
Monson et al.	Paramor Mimic	49.54	54.77	52.02
Our Algorithm	REC-TRIE	53.40	43.06	47.68
Lavallée and Langlais	RALI-COF	48.43	44.54	46.40
–	Morfessor CatMAP	79.38	31.88	45.49
Spiegler et al.	PROMODES 2	35.36	58.70	44.14
Spiegler et al.	PROMODES	32.22	66.42	43.39
Bernhard	MorphoNet	61.75	30.90	41.19
Can and Manandhar	2	41.39	38.13	39.70
Spiegler et al.	PROMODES committee	55.30	28.35	37.48
Golénia et al.	UNGRADE	46.67	30.16	36.64
Virpioja and Kohonen	Allomorfessor	85.89	19.53	31.82
–	Morfessor Baseline	89.68	17.78	29.67
Lavallée and Langlais	RALI-ANA	69.52	12.85	21.69
–	Letters	8.66	99.13	15.93
Can and Manandhar	1	73.03	8.89	15.86

and “adInIz” is segmented from adI since the most occurred character in these paths are I with 5293 occurrence (b). Next REC-TRIE constructs affix tries to find affix boundaries. In (c), affixes are inserted in a new trie. Note that “a” is merged from ab+a and ad+a and occurrence is summed. Again first affixes are found by selecting the most occurred character. This procedure continues until there is no affix candidate left. The final affix is found in (d) and REC-TRIE finish segmentation.

4 Experiments

Morpho Challenge gives two perl scripts to evaluate algorithms. These scripts simply compare the results against a linguistic gold standard. In the evaluation, only a subset of all words in the corpus is included. For each language, a random subset was picked, and the segmentations of these words were compared to the reference segmentations in the gold standard. The evaluation of an algorithm is based on the F-measure, which is the harmonic mean of precision and recall.

These metrics are calculated by

- Hit (H): The word is cut at the right place.
- Insertion (I): The word is cut at the wrong place.
- Deletion (D): A valid cut is ignored.

Based on these metrics; precision is the number of hits divided by the sum of the number of hits and insertions, and recall is the number of hits divided by the sum of the number of hits and deletions. So the measures are:

$$Precision = \frac{H}{(H+I)} Recall = \frac{H}{(H+D)} F - Measure = \frac{2H}{(2H+I+D)} \quad (1)$$

We have used the dataset of the Morpho Challenge 2009 and evaluate results with the perl scripts provided. Table 1 show the results of our algorithm compared with other algorithms for Turkish. Our algorithm has better F-Measure in Turkish and English than Finnish. This is due to fact that our algorithm finds roots and suffixes by traversing the trie one character at a time so found roots and suffixes are generally short. However, Finnish root words are rather long in the average due to the conservativeness of the language. Especially deletions are fairly more in Finnish since the algorithm oversegments the words. As a result, recall values for Finnish is low and pulls down the F-Measure dramatically.

5 Conclusions

We propose a novel algorithm for unsupervised morphological analysis, based on trie data structure and word occurrences. The algorithm constructs a word trie and finds root words according to the occurrences of characters in the path. After root detection is completed, remaining affix parts are used to construct affix tries. In each iteration, affix boundaries are detected and results are updated.

Although our proposed algorithm is simple, the results are encouraging. Our approach ranks 4'th in Turkish when compared with other competitors. The recall values states that we have missed some of the boundaries especially for Finnish.

The algorithm does not have any methods for prefix detection and there is no control for the irregular changes of the words or umlauts, so we should develop some strategies to cope with these situations.

References

1. Kurimo, M., Krista Lagus, S.V., Turunens, V.: Morpho challenge. http://research.ics.tkk.fi/events/morphochallenge2010/
2. Bernhard, D.: Unsupervised morphological segmentation based on segment predictability and word segments alignment. In: Proceedings of the PASCAL Challenge Workshop on Unsupervised Segmentation of Words into Morphemes (2006)
3. Keshava, S.: A simpler, intuitive approach to morpheme induction. In: Proceedings of the PASCAL Challenge Workshop on Unsupervised Segmentation of Words into Morphemes (2006)
4. Zeman, D.: Unsupervised acquiring of morphological paradigms from tokenized text. Adv. Multiling. Multimodal Inf. Retr. **5152**, 892–899 (2008)
5. Monson, C., Carbonell, J., Lavie, A., Levin, L.: Paramor and morpho challenge 2008. In: Proceedings of the 9th Cross-language evaluation forum conference on evaluating systems

for multilingual and multimodal information access. Cross-Language Evaluation Forum'08, pp. 967–974 (2009)
6. Monson, C., Hollingshead, K., Roark, B.: Probabilistic ParaMor. In: Working Notes for the CLEF 2009 Workshop, Corfu, Greece (2009)
7. Solak, A., Oflazer, K.: Design and implementation of a spelling checker for turkish. In: Literary and Linguistic Computing, vol. 8 (1993)