Subword Language Modeling Using Morphological Units Induced from Lexicon Automata

Anton Ragni
Department of Physics
University of Tartu
51010 Tartu, Estonia
ragni@ut.ee

Abstract

Subword language modeling is a practical must for automatic speech recognition of inflective and agglutinative languages. Algorithms developed for unsupervised induction of subword units from large text corpora either can not be applied to the target languages or do not address desirable characteristics of such units. In this paper we describe recently developed algorithm that meets all needs of subword language modeling. Preliminary experiments show that at least in terms of perplexity this algorithm outperforms a baseline model and gives a substantial reduction of unknown words.

1 Introduction

Word–based modeling of several languages is inappropriate for such tasks like automatic speech recognition (ASR). These languages usually belong to the group of inflective and agglutinative languages. For any given word as a rule the number of possible word forms is large. Additional words can be constructed by gluing basic word forms together. It makes the number of distinct words in such languages potentially infinite.

Estonian is a particular example of inflective and agglutinative language. The basic statistics from Estonian text corpora shows that approximately each two words out of hundred are new ones. If you consider an average corpus consisting of 100 millions of words then the number of distinct words can approach as much as 2 millions.

For languages belonging to different groups like English, German etc. this number usually does not exceed 60,000 of words. Moreover, it fits well into the upper limit imposed by the popular and compact way of indexing such words using 2–byte integers ($2^{16} = 65,536$). If someone tries to pursue the same strategy for languages like Estonian then the number of unknown words can contribute up to 10% to the overall number of wrongly recognized words.

One possible approach to get around this problem consists of splitting words into a number of smaller parts. The splitting procedure can be motivated either linguistically or mathematically. In the former case these subword units correspond to prefixes, stems etc. In the latter case they usually allow to achieve the best compression effect of training corpus. In both cases it is common to refer to them as *morphs*.

Linguistically motivated approaches are usually tailored to a particular group of languages. For example, (Goldsmith, 2000) considers European languages conforming only to *stem+suffix* structure. Mathematically motivated approaches are usually defined in a probabilistic framework and aim at finding segmentations which maximize or minimize some objective quantity. For example, (Creutz and Lagus, 2002) introduce an algorithm which makes segmentations by minimizing the cost required to represent them in a corpus.

The main problem with approaches introduced so far originates from the fact that they were developed keeping in mind extraction of trully morphological information from words. These morphemes, which in linguistics are usually defined as a smallest–meaning bearing units of language, are not constrained and can be as small as a single letter. From the acoustical point of view, discrimination of such morphemes is hard and confusable. In ASR, one is usually aimed at discovering much longer units to aid to the accurate recognition.

The rest of the paper is organized as follows. A brief description of algorithm is given in Section 2. Section 3 describes some language modeling experiments we performed trying to make them maximally close to subsequent application in speech recognition. Section 4 discusses some shortcomings we have encountered trying to apply such language models in a speech recognition task and suggests some possible remedies to them.

2 Algorithm

The algorithm for unsupervised induction of morphology from large text corpora makes use of finite–state automata framework used widely in natural language processing (NLP). Morphological analyzers, language models and word lattices are common examples of such application.

2.1 Description

The algorithm encodes the entire training corpus in a single finite–state automaton. Just in the same way as it is used in NLP for representing large dictionaries (Mohri, 1996). However, the same automaton can be also used to discover morphology if we consider the number of outgoing transitions from any given state as a *morpheme boundary indicator*. When the number of different transitions is large enough then there is a certain confidence that this state separates distinct morphemes from each other. One morpheme encoded prior to this state and others beginning on the outgoing transitions. Moreover, this confidence is dictated by the language itself – the more representative is training corpus the higher confidence will be.

If each transition in addition to a label also bears a numeric quantity describing how often it is traversed during the composition of automaton, then this number can be used to force introduction of unreliable morphemes. Otherwise such morphemes will be introduced in case of few "noisy" words. Misspelt, damaged, artefact and other "words" contribute to the source of possible errors.

2.2 Example

Consider a list of English words in the first column of Table 1 which is extracted from an imaginary text corpus. Assume further we have information how

Word	Segmentation
affect 1	affect
affecting 1	affect + ing
affectingly 1	affect + ing + ly
affection 1	affect + ion
affectionate 1	affect + ion + ate
affections 1	affect + ion + s
affects 1	affect + s

Table 1: List of English words with segmentations produced by recursive minimum description length method

frequently each word occurs in the corpus. If we encode the list into finite–state automaton then its graphical representation can be as the one given by Fig 1 (by the moment we leave weights out of consideration).

If we pursue the same strategy as described in Section 2.1 and additionally impose a restriction on the minimal length of morpheme to be at least two letters then morphological segmentations produced by the algorithm will be equal to those given in the second column of Table 1. Interestingly that the same result is obtained by a mathematical algorithm which aims at finding segmentations by minimizing a description length of lexicon and training corpus (Creutz and Lagus, 2002). From linguistical point of view these segmentations are almost perfect except for the prefix *af*.

Note that despite on the minimal morpheme length which equals in our example to two letters the segmentations of Table 1 still contain a single letter ending *s*. In ASR it is preferable to avoid such short morphemes. So we need to apply a constraint at word endings since final states of automaton will terminate a morpheme with no regard to any criteria.

2.3 Segmentation Accuracy

Accuracy of this algorithm was evaluated in the task of segmenting gold standard data. The gold standard

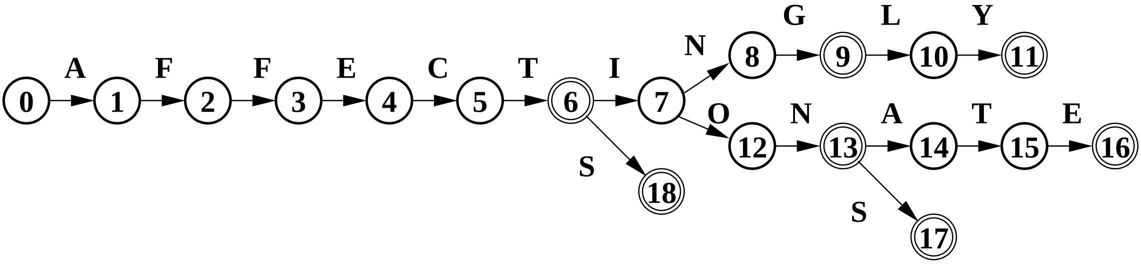

Figure 1: Example of representing lexicon using finite–state automaton. Encoded words are given in Table 1. Initial state is denoted by 0, end states by double circles. Transition weights are present but not shown.

data contains 40,000 manually corrected segmentations first preprocessed by a morphological analyzer. Evaluation results showed that in a basic configuration the algorithm attains a precision and recall of 64 and 30%. Despite on a such small accuracy of segmentation this result is still better than 83 and 4% showed by the baseline model since high precision is completely washed off by a negligible number of correctly discovered boundaries. A more comprehensive description of these experiments will shortly appear in (Ragni, 2007a).

3 Experiments

In this section we compare the algorithm just described with a baseline model in a language modeling task. The training corpus is given to both algorithm to produce morphological segmentations. These segmentations is used to rewrite the training corpus. The modified training corpus is used to create n–gram language models. The development set rewritten using the same segmentations is used as evaluation data. Here we assume that model giving the lowest perplexity on the evaluation data will be used in a subsequent speech recognition experiment. Therefore we use perplexity as the evaluation measure. However, one should be always aware of the fact that correlation between perplexity and accuracy of recognition is weak. This means that the lowest perplexity does not necessarily mean the highest accuracy. Nevertheless, the lowest perplexity is a good prerequisite of such.

3.1 Experimental Setup

The mixed corpus of Estonian (MCE) collected and maintained by the Computer Linguistics Group at University of Tartu[1] is used as the training corpus in this study. MCE primarily consists of articles from local newspapers and magazines. The total number of words is approximately 77 millions among which 1.7 million of words are distinct.

As a baseline model in this study we use publicly available Morfessor software (Creutz and Lagus, 2005). Morfessor derives morphological segmentations by minimizing the description length of training corpus. The description length is given as a cost in bits required to code training data. Once initial segmentations have been produced Morfessor iteratively resegments the corpus until no further improvement (in bits) is gained between two successive iterations.

Both algorithms use MCE to produce morphological segmentations. There are approximately 400,000 distinct morphs in each set of segmentations. A large number of morphs in these sets can be rewritten using the remaining morphs. Each morph having a length of at least five letters is checked whether it can be split into smaller parts. The size of morph lexicon is further reduced by cutting off morphs with small frequency of occurrence. Final lexicons for both algorithms contain at most 65,000 items. A special tag `<w>` is used in the training corpus to denote word boundaries and to allow reconstruction of words in the future output of speech recognizer.

For language model building we use the SRILM toolkit of (Stolcke, 2002) which allows to create n–gram models of arbitrary order with different probability smoothing techniques. In this study we constrain ourselves to fourgram language models with Linear, Good–Turing, Witten–Bell and Kneser–Ney

[1] Available on–line from http://www.cl.ut.ee

(original and modified) smoothing.

3.2 Results

Transcriptions from the development set of Babel speech database are used to assess the performance of n–gram models. Table 2 shows perplexities for both algorithms using different approaches to probability smoothing. In both cases the smallest perplexity is obtained using original Kneser–Ney discounting. Evaluation results show that fourgram language model built on top of segmentations produced by finite–state automaton has smaller perplexity than the baseline model. The number of unknown words is kept behind the level of 1% which addresses the shortcoming of word–based language models having OOV rate more than 10% (Ragni, 2007b). Table 3 gives n–gram access statistics. Except for bigram case the overall hit–ratio is higher for n–gram models based on the new approach.

Smoothing	**Perplexity**	
	MF	**LA**
Linear	59.9	40.9
Good–Turing	57.1	39.4
Witten–Bell	56.9	39.0
ΔKneser–Ney	55.1	38.0
Kneser–Ney	53.7	37.2
OOV rate	4.7%	0.86%

Table 2: Development set perplexities (PP) and out–of–vocabulary (OOV) rates for fourgram language models based on Morfessor (MF) and Lexicon Automaton (LA) algorithms

Order	**Hit–ratio**	
	MF	**LA**
n=2	97.1	95.7
n=3	50.6	58.6
n=4	56.6	64.5

Table 3: N–gram hits for language models based on Morfessor and Lexicon Automaton algorithms

4 Discussion

At least one important aspect needs to be discussed here if one tries to use subword language models described here. In order to make reconstruction of words possible at the output of speech recognizer we append each word segmentation with the boundary tag `<w>`. The assumption here is that a single–state non–emitting HMM model can be set into a correspondence with it. Some decoders like a large vocabulary recognizer in the HTK[2] toolkit are tailored to use a short–pause model to address possible periods of silence between words. Label of short–pause model does not appear at the output of recognizer and two consecutive skip models are not allowed in a search tree. To overcome this problem a different decoder may be used or the reconstruction process can be modified to use hyphenation marks instead of a single boundary tag. The latter approach however increases the lexicon size since the same morphological unit can appear separately or in the context of complex word. For example, Estonian morph `aja` may appear in the lexicon as `aja` and `aja-`. In the latter case the hyphen mark is used to indicate that the following morph should be tied with `aja`.

References

[Creutz and Lagus2002] M. Creutz and K. Lagus. 2002. Unsupervised discovery of morphemes. In *SIGPHON*, pages 22–30, Philadelphia.

[Creutz and Lagus2005] M. Creutz and K. Lagus. 2005. Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0. Technical Report Publications in Computer and Information Science, Report A81, Helsinki University of Technology.

[Goldsmith2000] J. Goldsmith. 2000. Unsupervised learning of the morphology of a language. *Computational Linguistics*, 27(2):153–198.

[Mohri1996] M. Mohri. 1996. On some applications of finite-state automata theory to natural language processing. *Natural Language Engineering*, 2(1):61–80.

[Ragni2007a] A. Ragni. 2007a. Inducing morphological units from lexicon automaton (submitted). In *Proceedings of 3rd Baltic Conference on HLT*, Kaunas.

[Ragni2007b] A. Ragni. 2007b. Initial experiments with estonian speech recognition. In *Proceedings of 16th Nordic Conference of Computational Linguistics*, Tartu.

[Stolcke2002] A. Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proceedings of ICSLP*, pages 901–904, Denver, USA, September.

[2] Available online from http://htk.eng.cam.ac.uk/