MORPHEME-BASED LANGUAGE MODELING FOR ARABIC LVCSR

*Ghinwa Choueiter**

MIT CS and AI Laboratory
32 Vassar St., Cambridge, MA 02139
ghinwa@mit.edu

Daniel Povey, Stanley F. Chen, and Geoffrey Zweig

IBM T. J. Watson Research Center
Yorktown Heights, NY 10598
{dpovey,stanchen,gzweig}@us.ibm.com

ABSTRACT

In this paper, we concentrate on Arabic speech recognition. Taking advantage of the rich morphological structure of the language, we use morpheme-based language modeling to improve the word error rate. We propose a simple constraining method to rid the decoding output of illegal morpheme sequences. We report the results obtained for word and morpheme language models using medium (<64kw) and large (∼800kw) vocabularies, the morpheme LM obtaining an absolute improvement of 2.4% for the former and only 0.2% for the latter. The 2.4% gain surpasses previous gains for morpheme-based LMs for Arabic, and the large vocabulary runs represent the first comparative results for vocabularies of this size for any language. Finally, we analyze the performance of the morpheme LM on word OOV's.

1. INTRODUCTION

The Arabic language is characterized by a complex morphological structure. It is, in fact, a highly inflected language where prefixes and suffixes are appended to the stem of a word to indicate tense, case, gender, number, etc. Hence, it is natural that this leads to rapid vocabulary growth which is accompanied by worse language model (LM) probability estimation and a higher out-of-vocabulary (OOV) rate. One would suspect that words are not the best lexical units in this case and, perhaps, sub-word units would be a better choice. In this work, we choose morphemes as our sub-word units and we generate them using an LM-based approach [1].

Although there has been much research on Arabic speech recognition such as the IBM® ViaVoice™[2] and BBN Tides-OnTap systems[3], there has not been much work studying Arabic morpheme-based language modeling. Apart from the 2002 JHU research on Arabic recognition [4] and the related work of Vergyri *et al.* [5], most research on morpheme-based systems has been developed for other inflected languages such as Turkish[6], German[7], Russian[8], and Korean[9]. In [4], the authors report an improvement of 0.7% absolute WER by combining a word model with a morpheme-based model. Vergyri *et al.* [5] report a gain of up to 1.5% absolute by utilizing morphological information within a factored LM.

Arabic Word	Morphological Decomposition	Meaning
WB_RXSXAAMX	WB_RXSXAAMX	painter
WB_AALXRXSXAAMXUXNX	WB_AALX# RXSXAAMX +UXNX	the painters
WB_RXSXAAMXIXNX	WB_RXSXAAMX +IXNX	two painters
WB_LXLXRXSXAAMXAATX	WB_LXLX# RXSXAAMX +AATX	to the painters (feminine)

Table 1. Different variations on the word *RXSXMX: paint* and the corresponding morpheme decomposition and meanings.

In this paper, we concentrate our efforts on Modern Standard Arabic, where we build morpheme-based LMs and study their effect on the OOV rate as well as the WER. In Section 2, we describe the morpheme-based language modeling used in our experiments. In Section 3, we describe the Arabic data sets used for training, testing, and building the LMs. In Section 4, we describe the experimental setup. In Section 5, we give the results and we discuss the difference in OOV rates for the word and morph models. We conclude in Section 6.

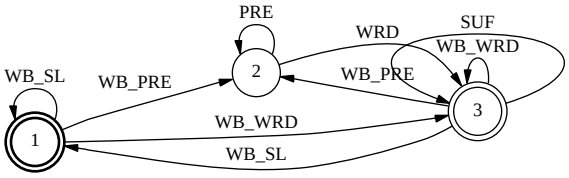

Fig. 1. The morpheme lattice constrainer

2. MORPHEME-BASED LANGUAGE MODELING

In order to build morpheme-based LMs, the data sets are decomposed into their morpheme components. An LM-based morpheme generator previously proposed in [1] is used to decompose the data. Every word is split into zero or more prefixes followed by a stem followed by zero or more suffixes.

*The first author performed this work while at IBM T.J. Watson.

For example,

$$w_1 w_2 w_3 \cdots \rightarrow m_1^1 m_1^2 \ldots m_1^i m_2^1 m_2^2 \ldots m_2^j m_3^1 m_3^2 \ldots m_3^k$$

where w_i's are the words and m_i's are the morpheme units which could be prefixes, stems, or suffixes. The units are then used to generate morpheme-based LMs:

$$P(m_1, m_2, \ldots, m_k) = \prod_{i=1}^{k} P(m_i | m_{i-N+1} \ldots m_{i-1}) \quad (1)$$

where N is the LM order.

Table 1 depicts the decomposition of sample variations of the word *RXSXAAMX: painter* [1] and illustrates the change in semantic meaning as different prefixes and suffixes are added to the stem. We append a # to the end of a prefix and a + to the beginning of a suffix. We also append the symbol WB_ to the beginning of each word prior to the morpheme decomposition. This, unfortunately, leads to an increase in morpheme vocabulary size and a distinction between a stem and the same stem preceded by a prefix as shown in the first and second rows of Table 1. However, this is needed to guarantee that the word boundaries are placed at the right location in the decoding graph for purposes of acoustic context expansion.

2.1. Morpheme Lattice Constrainer

With the morpheme LM comes the problem of decoding illegal morpheme sequences which result in outputs that are non-words. For example, consider the following sequence:

$$WB\_PRE_1 \quad WB\_PRE_2 \quad WRD_1 \quad SUF_1 \quad WB\_WRD_2 \quad WRD_3$$

The sequence has two illegal occurrences. The first is the presence of two consecutive word boundary prefixes, and the second is the presence of a non-boundary word at the end although it is not preceded by a boundary prefix. In order to solve this problem, we propose a simple solution: constraining the morpheme lattices. Figure 1 illustrates a finite state acceptor which we will refer to as the *constrainer*. The reason for that is clear from the figure. The acceptor only allows legal sequences of morphemes which consist of zero or more silence tokens followed by zero or more prefixes, the first of which should be a word-boundary prefix, followed by a stem (WRD) followed by zero or more suffixes. The next step is to tag every token in the morpheme lattice as *WB_PRE, PRE, WRD, WB_WRD, SUF,* or *WB_SL*, and finally the constrainer is composed with the morpheme lattices to remove the illegal sequences. The acceptor can also be used to filter out an unwanted set of words, for example, foreign words, if required, by tagging them *ILL* for illegal.

[1] Arabic symbols are represented here with a two-letter Roman code.

	RT'04 (training)	Arabic Gigaword
# Utterances	∼6000	∼14000000
Word Vocab Size	43260	847935
Word OOV Rate	10.8%	1.43%
Morpheme Vocab Size	20125	811624
Morpheme OOV Rate	3.6%	0.33%

Table 2. Description of the two corpora used for language modeling.

3. DATA SETS

We use two corpora in our experiments. First, for purposes of training and testing, we use the RT'04 corpus, which consists of Arabic broadcast news. It contains 82 hours of training data and 1.2 hours of test data. Next, to build the LMs, we use the RT'04 (train) and Arabic Gigaword corpora described in more detail in Table 2. The Gigaword corpus is comprised of Arabic newswire text data collected from four sources: Agence France Presse, Al Hayat News Agency, Al Nahar News Agency, and Xinhua News Agency. All of the RT'04 (train) and a portion of the Gigaword corpus were used in our experiments.

Both data sets are in Arabic and words are the basic lexical units. To facilitate text manipulation, the data is romanized, *i.e.*, represented using roman characters. A simple mapping of each Arabic letter to a 2-roman-letter code is performed. The Arabic alphabet consists of 29 letters, 3 of which are vowels (*alif, waw, yaa'*) and one of which is a glottal stop (*hamza*). A *hamza* can also be combined with the vowels to give glottalized vowels. In the case of *alif*, a *hamza* can appear above or below the letter. We normalize the Arabic data such that several representations of the *alif* at the beginning of a word such as those including the *hamza* are mapped into one roman code. Diacritics that appear below or above a consonant representing short vowels, consonant doubling, or consonant stressing are discarded.

4. EXPERIMENTAL SETUP

We construct word and morpheme-based setups in order to compare their performances for medium and large vocabularies. We use a 3-gram word LM and a 7-gram morpheme LM. All of the LMs use modified Kneser-Ney smoothing [10]. The word and morpheme LMs are built for three different purposes: to generate medium vocabulary and large vocabulary lattices, and to rescore these lattices.

To generate the medium vocabulary lattices, we use an LM built with the RT'04 (train) corpus. To create the large vocabulary lattices, we use RT'04 (train) to build one LM and the Gigaword corpus (divided into four parts due to memory constraints) to build 4 LMs. The 5 LMs are then interpolated. Due to limitations of our decoder, the LMs had to be heavily pruned for lattice generation purposes. In the case of the morpheme LM, the order was reduced to 6 and a pruning thresh-

	Word	Morph
Initial	35.6%	36.1%
Constraining	—	34.8%
Rescoring	31.7%	29.6%
Constraining	—	29.3%

Table 3. WER for the medium vocabulary setup.

old of 10^{-7} and cutoff values of 0,1,2,2,2,2 were used (*i.e.*, bigrams occurring once or less were pruned, trigrams occurring twice or less were pruned, etc.). In the case of the word LM, a pruning threshold of 10^{-7} was used. Finally, for lattice rescoring purposes, the word and morpheme LMs were built with no pruning whatsoever.

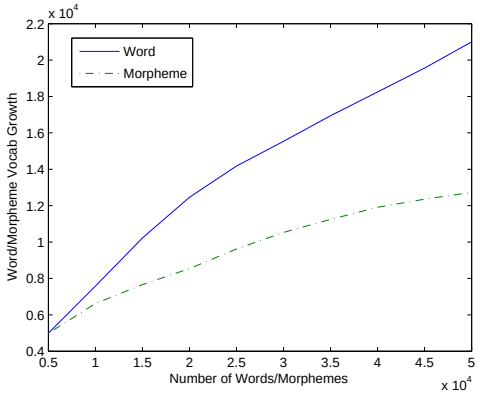

Fig. 2. Growth of the word and morpheme vocabularies.

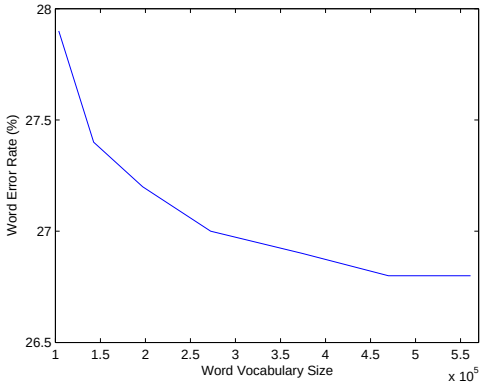

Fig. 3. WER of the word-based recognizer.

5. EXPERIMENTAL RESULTS

First, to get an idea of how word and morpheme vocabulary growth compare for our particular data sets, we generated Figure 2 using data from the Arabic Gigaword corpus. Figure 2 illustrates the growth in the number of unique Arabic words and morphemes as a function of corpus size. The plot shows a rapid vocabulary growth for words as compared to

	Word	Morph
Initial	28.3%	30.8%
Rescoring	27.1%	27.1%
Constraining	—	26.9%

Table 4. WER for the large vocabulary setup.

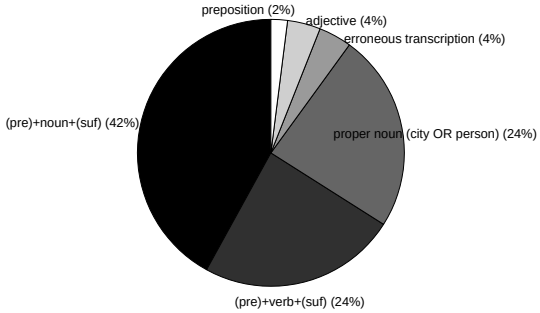

Fig. 4. The distribution of 50 randomly-selected OOV words. The label (pre)+noun+(suf), for example, implies zero or more prefixes, a noun, and zero or more suffixes.

morphemes. Also, Figure 3 illustrates the drop in WER as a function of vocabulary size in words. The plot is generated using the Arabic Gigaword corpus but with a different setup than the one used in our other experiments.

5.1. Word Error Rates

Table 3 shows the WER results for the medium vocabulary setup. The first row shows the results obtained for the highest-scoring paths in the original lattices obtained using the pruned medium vocabulary LMs. The second row shows the result for the morpheme-based system upon constraining the lattices. The third row shows the results after rescoring the lattices with the large LMs built using the Arabic Gigaword corpus. We observe a significant improvement as compared to the initial results. Finally, we constrain the rescored morpheme lattices and obtain our best result for the medium vocabulary system. We notice that the constraining factor has a smaller effect when applied to the rescored lattices than when applied to the original ones. This makes sense, since the larger the morpheme LM, the smaller the probabilities assigned to illegal sequences and the less likely they occur.

Type	# of Occurrences	# Recognized
(pre)+noun+(suf)	17	8
(pre)+verb+(suf)	9	2
proper name (person)	3	1
adjective	2	0
preposition	1	1

Table 5. The distribution of the 32 OOV words that can be recognized by the morpheme-based recognizer as well as the number of words recognized for the medium vocabulary case.

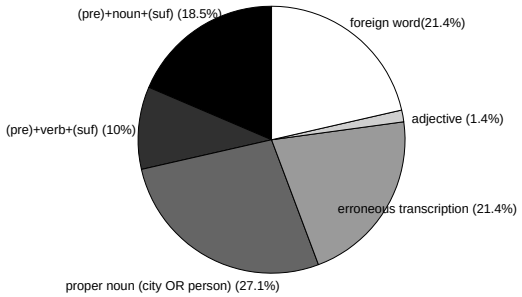

Fig. 5. The distribution of 70 OOV words obtained with the large vocabulary.

Type	# of Occurrences	# Recognized
(pre)+noun+(suf)	12	0
(pre)+verb+(suf)	5	1
proper name (person)	4	0
foreign	6	1

Table 6. The distribution of the 27 OOV words that can be recognized by the morpheme-based recognizer as well as the number of words recognized for the large vocabulary case.

Table 4 lists the WER results for the large vocabulary setup. Again, the first row shows the WER results for the original lattices obtained using the pruned large vocabulary LMs. The second row shows the results after rescoring the lattices with the large LMs. The last row gives the result of the morpheme-based system after being constrained. The improvement of the morpheme-based setup as compared to the word-based one is minimal in this case.

5.2. OOV Results

We first look at the out-of-vocabulary words for the medium vocabulary task. As mentioned in Section 3, the OOV rate for the word vocabulary is quite high at 10.8% while that of the morph is 3.6%. Comparing between the word and morph decoding outputs, we find that there are 761 unique OOV words of which 159 are decoded correctly at least once by the morpheme-based system. In order to get a better idea of the distribution of OOV words, we randomly select 50 samples. Figure 4 illustrates the different types of OOV's as well as their percentages. Of the 50 OOV words, 32 can be potentially recognized by the morpheme-based system, *i.e.*, their corresponding morphemes exist in the morpheme-based lattices. Of these 32, 12 are indeed recognized correctly (37.5%). Table 5 gives a description of the categories of these 32 words.

Next, we look at the large vocabulary task. We recall that OOV rates are much lower, 1.43% and 0.33% for the word and morpheme vocabularies, respectively. There are 70 unique OOV words, of which 27 can be recognized by the morpheme-based recognizer, and only 2 are. Figure 5 illustrates the distribution of all 70 OOV words.

6. DISCUSSION

We have proposed morpheme-based language modeling for Arabic speech recognition, and we have compared its performance against word-based models. For medium-sized vocabularies (<64kw), we achieve a 2.4% absolute WER improvement using a simple morpheme n-gram model, which compares favorably to the gains found in other morpheme-based work for Arabic [4, 5] while using a simpler model. In particular, we address the issue that more morphemes are needed to comprise the same LM context as compared to words by using a 7-gram model rather than a trigram. Previous work uses stream models, factored LMs, and/or combination with a word LM to incorporate morpheme information.

Much existing ASR software supports vocabularies of at most 64k words; thus, previous work with morpheme-based LMs (for any language) almost exclusively investigates vocabularies of this size or less. In this work, we show that the OOV issue for Arabic word LMs can largely be addressed by using a very large vocabulary and training set. With a vocabulary of 800k words, we found an OOV rate of 1.4% and morpheme LMs yielded a gain of only 0.2% absolute WER. A detailed analysis of the OOV words that morpheme LMs can potentially recognize reveal that most are foreign words, proper nouns, and erroneous transcriptions. It remains to be seen whether similar behavior holds for other highly-inflected languages.

7. REFERENCES

[1] Y. Lee et al., "Language model based Arabic word segmentation," in *ACL'03*, 2003, pp. 399–406.

[2] J.A. Baumgarten and K. Barksdale, *IBM® ViaVoice™ Quick Tutorial*, South-Western Educational Publishing, 2000.

[3] J. Billa et al., "Audio indexing of Arabic broadcast news," in *Proc. ICASSP '02*, Orlando, Florida, May 2002, vol. 1, pp. 5–8.

[4] K. Kirchhoff et al., "Novel speech recognition models with Arabic," *Johns-Hopkins University Summer Research Workshop, Final Report*, 2002.

[5] D. Vergyri, K. Kirchhoff, K. Duh, and A. Stolcke, "Morphology-based language modeling for Arabic speech recognition," in *Proc. ICSLP '04*, Jeju Island, Korea, Oct. 2004.

[6] K. Carki, P. Geutner, and T. Schultz, "Turkish LVCSR: Towards better speech recognition for agglutinative languages," in *Proc. ICASSP '00*, Istanbul, Turkey, June 2000, vol. 3, pp. 1563–1566.

[7] P. Geutner, "Using morphology towards better large-vocabulary speech recognition systems," in *Proc. ICASSP '95*, Detroit, MI, USA, May 1995, vol. 1, pp. 445–448.

[8] E. Whittaker and P. Woodland, "Particle-based language modelling," in *Proc. ICSLP '00*, Beijing, China, Oct. 2003.

[9] O.W. Kwon, "Performance of LVCSR with morpheme-based and syllable-based recognition units," in *Proc. ICASSP '00*, Istanbul, Turkey, June 2000, vol. 3, pp. 1567–1570.

[10] S. F. Chen and J. T. Goodman, "An empirical study of smoothing techniques for language modeling," *Technical Report, Harvard University, Computer Science Group*, Aug. 1998.