Abstract

Performing large vocabulary continuous speech recognition (LVCSR) for morphologically rich languages is considered a challenging task. The morphological richness of such languages leads to high out-of-vocabulary (OOV) rates and poor language model (LM) probabilities. In this case, the use of morphemes has been shown to increase the lexical coverage and lower the LM perplexity. Another approach used to improve the LM probability estimates is to incorporate additional knowledge sources in the LM estimation process using class-based LMs (CLMs). Recently, the hierarchical Pitman-Yor LMs (HPYLMs) have shown superiority over the modified Kneser-Ney (MKN) smoothed N-gram LMs in terms of both perplexity (PPL) and word error rate (WER) on word-based LVCSR tasks. In this paper, hierarchical Pitman-Yor class-based LMs (HPY-CLMs) are combined with morpheme level language modeling. This enables the application of the proposed models on top of morpheme-based systems. Experiments are conducted on Arabic and German LVCSR tasks. Consistent performance improvements are obtained for all the available corpora compared to the conventional morpheme-based and class-based LMs.

Index Terms: language model, morpheme-based, class-based, hierarchical Pitman-Yor, rich morphology

1. Introduction

Arabic and German are characterized by a complex morphological structure. Arabic belongs to the family of Semitic languages. It is in fact a highly inflected language having a large number of different surface forms. The Arabic words are derived from roots, by applying patterns to get stems and then attaching different affixes to obtain a large number of word forms. Thus, a stem can be thought of as being further decomposed into a root and a pattern [1]. On the other hand, German belongs to the family of Germanic languages. It is also cited as an outstanding example of highly inflected languages, as a large number of words can be derived from the same root. In addition, German makes a liberal use of noun compounding. Also, the meaning of German words can be expanded through the use of prefixes [2]. This huge lexical variety of Arabic and German causes data sparsity problems and leads to high OOV rates and poor LM probability estimates indicated by high LM perplexities. Normally, a conventional Arabic LVCSR system uses a very large LM training corpora and recognition vocabulary. Yet, still relatively high WERs are observed.

An alternative approach to deal with morphological richness is to use sub-lexical LMs [3, 4]. Typically, morpheme-based LMs are used to reduce data sparsity, lower the OOV rate and perplexity, and thereby achieve lower WERs. Morpemes are the smallest linguistic components of the word that hold semantic meanings. They are generated by applying morphological decomposition to words based on supervised or unsupervised approaches. The supervised approaches make use of linguistic knowledge like in [5]. Other supervised methods rely on carefully built morphological analyzers like in [6, 7, 8]. On the other hand, the unsupervised approaches are statistical data-driven approaches like in [9, 10]. Other unsupervised methods are based on the minimum description length (MDL) principle like in [11]. On the contrary, the unsupervised approaches do not require any language specific knowledge.

Another approach to overcome the data sparseness and reduce the dependence of the word-based LMs on the discourse domain, is to assign proper features (classes) to words and build LMs over those features. This yields better smoothing and, hopefully, better generalization to unseen word sequences. The features can be generated based on linguistic methods [12], or via data-driven approaches [13]. One approach for incorporating word features into LMs is the class-based LM (CLM) [14]. It combines the N-gram model over classes with the probability distribution of words in classes in order to better estimate smoothed probabilities of word sequences. This type of LM can be used to perform N-best list rescoring.

Recently, there has been a considerable amount of research aimed at improving the fundamental modeling of the N-gram LMs. Among this, hierarchical Bayesian LMs [15] have succeeded to achieve a comparable performance to the state-of-the-art N-gram LMs smoothed with MKN smoothing. A hierarchical Pitman-Yor LM (HPYLM), initially introduced in [16], is a type of Bayesian LM based on the Pitman-Yor (PY) process that has been shown to improve the perplexity over the MKN smoothed N-gram LM. In [17], the HPYLM has been implemented as an extension to the SRILM toolkit [18] and WER improvements have been reported for typical LVCSR tasks.

This paper presents a novel approach that attempts to combine the benefits of all the aforementioned techniques. We make use of the HPYLM methodology to build CLMs using classes assigned on morpheme level. This is called morpheme level HPYCLM. Thereby, we gain the advantages of using morpheme-based LMs, along with the benefits of feature-rich modeling, in addition to the improvement from the HPYLM. Moreover, linear interpolation is performed to combine different types of LMs. The results are compared to our best previously published results in [19, 20]. Although little improvements are achieved over the best previous results, they are systematically consistent over all the available corpora.

Morpheme Level Hierarchical Pitman-Yor Class-based Language Models for LVCSR of Morphologically Rich Languages

Amr El-Desoky Mousa1, M. Ali Basha Shaik1, Ralf Schlüter1, Hermann Ney1,2

1Human Language Technology and Pattern Recognition, Computer Science Department, RWTH Aachen University, Aachen, Germany
2Spoken Language Processing Group, LIMSI CNRS, Paris, France

{desoky, shaik, schluter, ney}@cs.rwth-aachen.de
2. Word decomposition and class derivation

2.1. Arabic

The Arabic LM training data is processed using MADA 2.0 tool [21]. MADA is a morphological analyzer and disambiguation tool developed for Arabic language. It is built on top of the Buckwalter Arabic morphological analyzer (BAMA) [22]. It is able to associate a complete set of morphological tags with each word in context. These tags are used to produce robust word diacritization and tokenization. Based on this tokenization, we produce decomposed words in the form of “prefix + stem + suffix”, where the existence of the prefix and the suffix is optional. The ‘+’ sign is used as a marker for full-word recombination. For a detailed description of the decomposition process and constraints, see our previous work [7]. Moreover, in [7], it was found out that having around 20k most frequent decomposable full-words without decomposition (out of 250k items) in the recognition vocabulary is quite helpful to achieve high recognition performance.

Starting from the MADA morphological tags along with the generated decomposition, we derive two different features namely, “lexeme” and “morph”. Lexeme is an abstraction over the inflected words that groups together all word forms that differ only in one of the morphological categories such as number or gender. Morph is the morphological category of the word; it includes the word part-of-speech (POS) and indicates whether a conjunction, particle, article or a clitic are agglutinated to the word. In addition, a third feature called “pattern” is derived by subtracting root letters from the word. The root is generated by the “Sebawai” tool [23]. Similarly, these features can be defined for morphemes as well as for full-words. Thus, after performing word decomposition, lexeme; morph; and pattern features are assigned to the resulting morphemes separately. Furthermore, a feature vector is generated using the wAlSrypy tool running a data-driven classification algorithm. First, all the discrete vocabulary items are converted into real valued vectors using word-pair co-occurrence matrix and singular value decomposition (SVD) [27, 28], then these vectors are clustered into 250 clusters using a k-means approach. A detailed description of this algorithm is found in a previous publication [25].

Similar to Arabic, the LM training corpus is preprocessed so that every word or morpheme is replaced by a vector of features: \( W \in \{ w_{1}, w_{2}, ..., w_{M} \} \) is the full-gram, \( \text{baseform} \) is the canonical baseform of the word. The TreeTagger has been successfully used to tag words of many languages including German [26]. One of the useful properties of the TreeTagger is that it operates successfully over morphemes as well as full-words provided that the input morphemes are linguistically meaningful which is true in our case. In addition to POS and baseform, we derive a third feature called class-index. This is a data-driven class-index assigned to every word or morpheme after running a data-driven classification algorithm. First, all the discrete vocabulary items are converted into real valued vectors using word-pair co-occurrence matrix and singular value decomposition (SVD) [27, 28], then these vectors are clustered into 250 clusters using a k-means approach. A detailed description of this algorithm is found in a previous publication [25].

3. Morpheme level class-based LMs

Given a sequence of words \( W = w_{1}, w_{2}, ..., w_{M} \), a standard N-gram LM is expressed as:

\[
p(w_{1}, w_{2}, ..., w_{M}) \approx \prod_{i=1}^{M} p(w_{i} | w_{i-1:N-1})
\]  

If this model is built over decomposed words (morphemes), then it is called a morpheme level model. However, instead of building the N-gram LM over sequences of words \( W \), we could build the model over sequences of some selected class stream, like sequences of lexemes, morphs or patterns. The CLM, initially described in [14], aims at combining the N-gram model over classes with the probability distribution of words in classes in order to better estimate smoothed probabilities over word sequences. Assuming that ambiguous class membership is used (also called soft clustering), where a word can be a member of multiple classes, then a bigram CLM is given by Equation 2, where a word is denoted by \( w_{i} \) and \( c_{i} \) is the class assigned to the word \( w_{i} \) at time \( i \).

\[
p(w_{i} | w_{i-1}) = \sum_{c_{i-1}} p(w_{i} | c_{i}) p(c_{i} | c_{i-1}) p(c_{i-1} | w_{i-1})
\]  

It is stated in [24] that ignoring word counts in a given corpus and using only the corpus vocabulary to train the Morfessor model produces decompositions that almost resemble the linguistic morphemes. Therefore, we train our Morfessor model using a vocabulary of distinct words that occur more than 5 times in the training corpus. This gives about 0.5 Million words. Less frequent words are not included in training in order to avoid irregularities that are harmful to the training process. In addition, the resulting decompositions are modified to remove very short and noisy morphemes. The final set of morphemes appear linguistically meaningful, where the most dominant decompositions are mainly the decomposition of the compound words and the stripping off the common prefixes.

Word features are generated using the TreeTagger [26]. It is a probabilistic tool that uses decision trees for annotating text with part-of-speech (POS) and lemma information. Lemma is the canonical baseform of the word. The TreeTagger has been successfully used to tag words of many languages including German [26]. One of the useful properties of the TreeTagger is that it operates successfully over morphemes as well as full-words provided that the input morphemes are linguistically meaningful which is true in our case. In addition to POS and baseform, we derive a third feature called class-index. This is a data-driven class-index assigned to every word or morpheme after running a data-driven classification algorithm. First, all the discrete vocabulary items are converted into real valued vectors using word-pair co-occurrence matrix and singular value decomposition (SVD) [27, 28], then these vectors are clustered into 250 clusters using a k-means approach. A detailed description of this algorithm is found in a previous publication [25].

Similar to Arabic, the LM training corpus is preprocessed so that every word or morpheme is replaced by a vector of features: \( W \in \{ w_{1}, w_{2}, ..., w_{M} \} \) is the full-gram, \( \text{baseform} \) is the canonical baseform of the word. The TreeTagger has been successfully used to tag words of many languages including German [26]. One of the useful properties of the TreeTagger is that it operates successfully over morphemes as well as full-words provided that the input morphemes are linguistically meaningful which is true in our case. In addition to POS and baseform, we derive a third feature called class-index. This is a data-driven class-index assigned to every word or morpheme after running a data-driven classification algorithm. First, all the discrete vocabulary items are converted into real valued vectors using word-pair co-occurrence matrix and singular value decomposition (SVD) [27, 28], then these vectors are clustered into 250 clusters using a k-means approach. A detailed description of this algorithm is found in a previous publication [25].
An analogous model could be estimated for morphemes with properly assigned classes. In Equation 2, it can be seen that there are only two component distributions required to estimate the class-based probability. The first component is the probability distribution over sequences of classes (called class N-gram). The second component is the probability distribution of words given classes (called class membership definition).

Normally, the standard word-based N-gram LMs perform better in capturing the relations between words in in-domain text. Therefore, an effective way to retain the advantages of both word-based and class-based LMs is to combine them. The combination may rely on backing-off or linear interpolation [29].

Using the context of a unigram LM as in [16], let $W$ be a finite vocabulary of $V$ words. Let $G(w)$ be the probability of each $w \in W$, and let $G = \{G(w)\}_{w \in W}$ be the vector of word probabilities. We place a PY process prior on $G$:

$$G \sim PY(d, \theta, G_0)$$

where the parameters of the process are: a discount parameter $0 < d < 1$, a strength parameter $\theta > -d$, and a mean vector $G_0 = \{G_0(w)\}_{w \in W}$. $G_0(w)$ is the prior probability of word $w$, usually uniformly distributed, thus $G_0(w) = 1/V$.

Let $[x_i] = x_1, x_2, \ldots$ be a sequence of words drawn from $G$. The process can be described as a generative procedure that iteratively produces $[x_i]$ with $G$ marginalized out. This can be achieved by relating $[x_i]$ to another separate sequence of draws $[y_i] = y_1, y_2, \ldots$ from the mean distribution $G_0$ as follows. The first word $x_0$ is assigned the value of the first draw $y_1$ from $G_0$. Let $t$ be the current number of draws from $G_0$ (currently $t = 1$), $c_0$ be the number of words assigned the value of the draw $y_1$ (currently $c_1 = 1$), and $e = \sum_{i=1}^{t} c_i$ be the current number of draws from $G$. For each subsequent word $x_{t+1}$, we either assign it the value of a previous draw $y_t$ with probability $\frac{c_t}{e}$ (increment $c_t$; set $x_{t+1} \leftarrow y_t$), or we assign it the value of a new draw from $G_0$ with probability $\frac{e-t}{V}$ (increment $t$; set $c_t = 1$; draw $y_t \sim G_0$; set $x_{t+1} \leftarrow y_t$).

The above procedure is often referred to as the Chinese restaurant process [32]. Imagine a sequence of customers (corresponding to the draws from $G$) visiting a Chinese restaurant with an infinite number of tables (corresponding to the draws from $G_0$), each of which can accommodate an infinite number of customers. The first customer sits at the first available table, and each subsequent customer either joins an already occupied table (assign the word to a previous draw from $G_0$), or sits at a new table (assign the word to a new draw from $G_0$).

Now, an N-gram LM can be described as a hierarchical extension of the PY process. An N-gram LM defines probabilities over words given $N − 1$ context words. Given a context $u$, let $G_0(u)$ be the probability of the current word taking on value $w$. A PY process is used as a prior for $G_u = \{G_u(w)\}_{w \in W}$:

$$G_u \sim PY(d_{[u]}, \theta_{[u]}, G_0(u))$$

where $\pi(u)$ is the suffix context of $u$ after dropping the earliest word. We recursively place a prior over $G_{[u]}$ using 5, but with parameters $\theta_{[u]}$, and a mean vector $G_{[u]}$. This is repeated until reaching $G_\phi$ with an empty context, then a global uniform prior $G_\phi$ is placed on $G_\phi$, where $G_\phi(u) = 1/V$:

$$G_\phi \sim PY(d_\phi, \theta_\phi, G_\phi)$$

Starting from a posterior distribution over seating arrangements in a hierarchical Chinese restaurant, the predictive probability of a word $w$ after a context $u$ can be inferred using Gibbs sampling. A detailed inference scheme is described in [30].

5. Experimental setup

5.1. Arabic system

Arabic acoustic models (AMs) are triphone models trained on 1100h of audio material taken from two domains: broadcast news (BN) and broadcast conversation (BC). The basic AMs are trained using maximum likelihood (ML) method. Then, a discriminative training based on minimum phone error (MPE) criterion is performed to enhance the models. The LM training corpora have around 206 Million running words including data from Agile Arab text, FBIS, TDT4 and GALE BN and BC data. A morpheme-based system with a 250k vocabulary is used. The 20k most frequent full-words are preserved without decomposition [7]. The speech recognizer works in 3 passes. In the first pass, within-word AMs are used without adaptation. The second pass uses across-word AMs with constrained maximum likelihood linear regression (CMLLR) adaptation. Then, a third pass with maximum likelihood linear regression (MLLR) adaptation is performed. In each pass, a morpheme-based bigram LM is used to construct the search space and to produce lattices. Therefore, an effective way to retain the advantages of both word-based and class-based LMs perform better in capturing the relations between words in in-domain text. Therefore, an effective way to retain the advantages of both word-based and class-based LMs is to combine them. The combination may rely on backing-off or linear interpolation [29].

Using the context of a unigram LM as in [16], let $W$ be a finite vocabulary of $V$ words. Let $G(w)$ be the probability of each $w \in W$, and let $G = \{G(w)\}_{w \in W}$ be the vector of word probabilities. We place a PY process prior on $G$:

$$G \sim PY(d, \theta, G_0)$$

where the parameters of the process are: a discount parameter $0 < d < 1$, a strength parameter $\theta > -d$, and a mean vector $G_0 = \{G_0(w)\}_{w \in W}$. $G_0(w)$ is the prior probability of word $w$, usually uniformly distributed, thus $G_0(w) = 1/V$.

Let $[x_i] = x_1, x_2, \ldots$ be a sequence of words drawn from $G$. The process can be described as a generative procedure that iteratively produces $[x_i]$ with $G$ marginalized out. This can be achieved by relating $[x_i]$ to another separate sequence of draws $[y_i] = y_1, y_2, \ldots$ from the mean distribution $G_0$ as follows. The first word $x_1$ is assigned the value of the first draw $y_1$ from $G_0$. Let $t$ be the current number of draws from $G_0$ (currently $t = 1$), $c_0$ be the number of words assigned the value of the draw $y_1$ (currently $c_1 = 1$), and $e = \sum_{i=1}^{t} c_i$ be the current number of draws from $G$. For each subsequent word $x_{t+1}$, we either assign it the value of a previous draw $y_t$ with probability $\frac{c_t}{e}$ (increment $c_t$; set $x_{t+1} \leftarrow y_t$), or we assign it the value of a new draw from $G_0$ with probability $\frac{e-t}{V}$ (increment $t$; set $c_t = 1$; draw $y_t \sim G_0$; set $x_{t+1} \leftarrow y_t$).

The above procedure is often referred to as the Chinese restaurant process [32]. Imagine a sequence of customers (corresponding to the draws from $G$) visiting a Chinese restaurant with an infinite number of tables (corresponding to the draws from $G_0$), each of which can accommodate an infinite number of customers. The first customer sits at the first available table, and each subsequent customer either joins an already occupied
MKN smoothing via the SRILM toolkit. The second pass performs speaker adaptation based on both CMLLR and MLLR. A standard trigram LM is used to generate N-best lists (N=500), then N-best rescoring is performed using 4-gram conventional or trigram HPYCLMs using different classes as described in Section 2.2. The recognition performance is evaluated on the Quaero 2009 dev and eval corpora [dev09: 7.5h; eval09: 3.8h].

6. Experiments

Table 1 shows the baseline recognition results for word- and morpheme-based LMs on Arabic and German corpora after a conventional 4-gram LM lattice rescoring. It can be seen that significant improvements are achieved in WER using a morpheme-based LM compared to word-based LMs. On top of morpheme-based systems, Tables 2(a) and 2(b) present the recognition results after the final rescoring using MKN and HPY LMs built over different classes for both Arabic and German corpora. The column labeled “MKN” shows the results using a MKN smoothed N-gram LM interpolated with a MKN smoothed CLM using the available features as classes. In the last row of each table, the interpolation is extended to include all the CLMs built on all the available classes (1 N-gram LM + 3 CLMs). In a similar fashion, the column labeled “MKN+HPY” shows the results using an interpolation of MKN smoothed LM, a MKN smoothed CLM, a HPYLM, and a HPYCLMs. Similarly, in the last row, the interpolation is extended to include all the CLMs built on all the available classes (2 N-gram LMs + 6 CLMs). We can see that the best results are obtained when using the HPYCLMs with all the available classes. This means that both the use of multiple features and the application of HPY models are beneficial. Significant WER reductions of (dev07: 0.5% absolute (3.5% relative); eval07: 0.4% absolute (2.5% relative); dev09: 0.7% absolute (2.2% relative); eval09: 0.6% absolute (2.1% relative)) are achieved compared to the baseline morpheme-based systems in Table 1. Although little improvements in the recognition performance are achieved over the conventional CLMs (that do not use the HPY models), the improvements are quite persistent and systematically consistent over all the available corpora. The observed WER improvements are considered statistically significant using a bootstrap method of significance analysis described in [33], the probability of improvement (POI\textsubscript{boot}) ranges between 95% and 98%. It is worth noting that the character error rate (CER) improvements are almost going in-line with the WER improvements.

7. Conclusions

We have introduced a novel methodology that combines the benefits of: morpheme-based LMs, feature-rich CLMs, along with HPYCLMs to perform LVCSR for Arabic and German as examples of morphologically rich languages. The morpheme-based modeling aims at increasing the lexical coverage and reducing the data sparseness, while the use of morpheme level features in CLMs attempts to achieve better generalization to unseen word sequences. At the same time, the use of HPYCLMs improves the smoothness of the N-gram probabilities over the conventional MKN smoothing for both normal and class-based models. We used different types of morphological and data-driven features for building CLMs. The best results are achieved by interpolating all the normal and the class-based LMs together. Proper tests have shown the statistical significance of the obtained WER improvements compared to the conventional morpheme-based LMs alone. Little but systematically consistent improvements are achieved over the conventional CLMs.

8. Acknowledgments

This work was partly funded by the European Community’s 7th Framework Programme under the project SCALE (FP7-213850), and partly realized under the Quaero Programme, funded by OSEO, French State agency for innovation. Hermann Ney was partially supported by a senior chair award from DIGITEO, a French research cluster in Ile-de-France.
9. References


