Learning a syntagmatic and paradigmatic structure from language data with a bi-multigram model

Sabine Deligne and Yoshinori Sagisaka
ATR-ITL, dept1, 2-2 Hikaridai
Seika cho, Soraku gun, Kyoto fu 619-0224, Japan.

Abstract

In this paper, we present a stochastic language modeling tool which aims at retrieving variable-length phrases (multigrams), assuming bigram dependencies between them. The phrase retrieval can be intermixed with a phrase clustering procedure, so that the language data are iteratively structured at both a paradigmatic and a syntagmatic level in a fully integrated way. Perplexity results on ATR travel arrangement data with a bi-multigram model (assuming bigram correlations between the phrases) come very close to the trigram scores with a reduced number of entries in the language model. Also the ability of the class version of the model to merge semantically related phrases into a common class is illustrated.

1 Introduction

There is currently an increasing interest in statistical language models, which in one way or another aim at exploiting word-dependencies spanning over a variable number of words. Though all these models commonly relax the assumption of fixed-length dependency of the conventional ngram model, they cover a wide variety of modeling assumptions and of parameter estimation frameworks. In this paper, we focus on a phrase-based approach, as opposed to a gram-based approach: sentences are structured into phrases and probabilities are assigned to phrases instead of words. Regardless of whether they are gram or phrase-based, models can be either deterministic or stochastic. In the phrase-based framework, non determinism is introduced via an ambiguity on the parse of the sentence into phrases. In practice, it means that even if phrase abc is registered as a phrase, the possibility of parsing the string as, for instance, $[ab]$ $[c]$ still remains. By contrast, in a deterministic approach, all co-occurences of a, b and c would be systematically interpreted as an occurence of phrase $[abc]$.

Various criteria have been proposed to derive phrases in a purely statistical way [1]; data likelihood, leaving-one-out likelihood (Ries et al., 1996), mutual information (Suhm and Waibel, 1994), and entropy (Masataki and Sagisaka, 1996). The use of the likelihood criterion in a stochastic framework allows EM principled optimization procedures, but it is prone to overlearning. The other criteria tend to reduce the risk of overlearning, but their optimization relies on heuristic procedures (e.g. word grouping via a greedy algorithm (Matsunaga and Sagayama, 1997)) for which convergence and optimality are not theoretically guaranteed. The work reported in this paper is based on the multigram model, which is a stochastic phrase-based model, the parameters of which are estimated according to a likelihood criterion using an EM procedure. The multigram approach was introduced in (Bimbot et al., 1995), and in (Deligne and Bimbot, 1995) it was used to derive variable-length phrases under the assumption of independence of the phrases. Various ways of theoretically releasing this assumption were given in (Deligne et al., 1996). More recently, experiments with 2-word multigrams embedded in a deterministic variable ngram scheme were reported in (Siu, 1998).

In section 2 of this paper, we further formulate a model with bigram (more generally $\overline{n}$-gram) dependencies between the phrases, by including a paradigmatic aspect which enables the clustering of variable-length phrases. It results in a stochastic class-phrase model, which can be interpolated with the stochastic phrase model, in a similar way to deterministic approaches. In section 3 and 4, the phrase and class-phrase models are evaluated in terms of perplexity values and model size.

2 Theoretical formulation of the multigrams

2.1 Variable-length phrase distribution

In the multigram framework, the assumption is made that sentences result from the concatenation of variable-length phrases, called multigrams. The likelihood of a sentence is computed by summing the likelihood values of all possible segmentations of the sentence into phrases. The likelihood computa-

[1] i.e. without using grammar rules like in Stochastic Context Free Grammars.

tion for any particular segmentation into phrases depends on the model assumed to describe the dependencies between the phrases. We call **bi-multigram model** the model where bigram dependencies are assumed between the phrases. For instance, by limiting to 3 words the maximal length of a phrase, the bi-multigram likelihood of the string "a b c d" is:

$$\sum \begin{cases} p([a] \mid \#)\, p([b] \mid [a])\, p([c] \mid [b])\, p([d] \mid [c]) \\ p([a] \mid \#)\, p([b] \mid [a])\, p([cd] \mid [b]) \\ p([a] \mid \#)\, p([bc] \mid [a])\, p([d] \mid [bc]) \\ p([a] \mid \#)\, p([bcd] \mid [a]) \\ p([ab] \mid \#)\, p([c] \mid [ab])\, p([d] \mid [c]) \\ p([ab] \mid \#)\, p([cd] \mid [ab]) \\ p([abc] \mid \#)\, p([d] \mid [abc]) \end{cases}$$

To present the general formalism of the model in this section, we assume $\overline{n}$-gram correlations between the phrases, and we note n the maximal length of a phrase (in the above example, $\overline{n}$=2 and n=3). Let W denote a string of words, and $\{S\}$ the set of possible segmentations on W. The likelihood of W is:

$$\mathcal{L}(W) = \sum_{S \in \{S\}} \mathcal{L}(W, S) \tag{1}$$

and the likelihood of a segmentation S of W is:

$$\mathcal{L}(W, S) = \prod_{\tau} p(s_{(\tau)} \mid s_{(\tau-\overline{n}+1)} \cdots s_{(\tau-1)}) \tag{2}$$

with $s_{(\tau)}$ denoting the phrase of rank (τ) in the segmentation S. The model is thus fully defined by the set of $\overline{n}$-gram probabilities on the set $\{s_i\}_i$ of all the phrases which can be formed by combining 1, 2, ...up to n words of the vocabulary. Maximum likelihood (ML) estimates of these probabilities can be obtained by formulating the estimation problem as a ML estimation from incomplete data (Dempster et al., 1977), where the unknown data is the underlying segmentation S. Let $Q(k, k+1)$ be the following auxiliary function computed with the likelihoods of iterations k and $k+1$:

$$Q(k, k+1) = \sum_{S \in \{S\}} \mathcal{L}^{(k)}(S \mid W) \log \mathcal{L}^{(k+1)}(W, S) \tag{3}$$

It has been shown in (Dempster et al., 1977) that if $Q(k, k+1) \geq Q(k, k)$, then $\mathcal{L}^{(k+1)}(W) \geq \mathcal{L}^{(k)}(W)$. Therefore the reestimation equation of $p(s_{i_{\overline{n}}} \mid s_{i_1} \ldots s_{i_{\overline{n}-1}})$, at iteration $(k+1)$, can be derived by maximizing $Q(k, k+1)$ over the set of parameters of iteration $(k+1)$, under the set of constraints $\sum_{s_{i_{\overline{n}}}} p(s_{i_{\overline{n}}} \mid s_{i_1} \ldots s_{i_{\overline{n}-1}}) = 1$, hence:

$$p^{(k+1)}(s_{i_{\overline{n}}} \mid s_{i_1} \ldots s_{i_{\overline{n}-1}}) = \frac{\sum_{S \in \{S\}} c(s_{i_1} \ldots s_{i_{\overline{n}-1}} s_{i_{\overline{n}}}, S) \times \mathcal{L}^{(k)}(S \mid W)}{\sum_{S \in \{S\}} c(s_{i_1} \ldots s_{i_{\overline{n}-1}}, S) \times \mathcal{L}^{(k)}(S \mid W)} \tag{4}$$

where $c(s_{i_1} \ldots s_{i_{\overline{n}}}, S)$ is the number of occurences of the combination of phrases $s_{i_1} \ldots s_{i_{\overline{n}}}$ in the segmentation S. Reestimation equation (4) can be implemented by means of a forward-backward algorithm, such as the one described for bi-multigrams ($\overline{n} = 2$) in the appendix of this paper. In a decision-oriented scheme, the reestimation equation reduces to:

$$p^{(k+1)}(s_{i_{\overline{n}}} \mid s_{i_1} \ldots s_{i_{\overline{n}-1}}) = \frac{c(s_{i_1} \ldots s_{i_{\overline{n}-1}} s_{i_{\overline{n}}}, S^{*(k)})}{c(s_{i_1} \ldots s_{i_{\overline{n}-1}}, S^{*(k)})} \tag{5}$$

where $S^{*(k)}$, the segmentation maximizing $\mathcal{L}^{(k)}(S \mid W)$, is retrieved with a Viterbi algorithm.

Since each iteration improves the model in the sense of increasing the likelihood $\mathcal{L}^{(k)}(W)$, it eventually converges to a critical point (possibly a local maximum).

2.2 Variable-length phrase clustering

Recently, class-phrase based models have gained some attention (Ries et al., 1996), but usually it assumes a previous clustering of the words. Typically, each word is first assigned a word-class label "$< \mathcal{C}_k >$", then variable-length phrases $[\mathcal{C}_{k_1}\mathcal{C}_{k_2}...\mathcal{C}_{k_l}]$ of word-class labels are retrieved, each of which leads to define a phrase-class label which can be denoted as "$< [\mathcal{C}_{k_1}\mathcal{C}_{k_2}...\mathcal{C}_{k_l}] >$". But in this approach only phrases of the same length can be assigned the same phrase-class label. For instance, the phrases "thank you for" and "thank you very much for" cannot be assigned the same class label. We propose to address this limitation by directly clustering phrases instead of words. For this purpose, we assume bigram correlations between the phrases ($\overline{n} = 2$), and we modify the learning procedure of section 2.1, so that each iteration consists of 2 steps:

- **Step 1** Phrase clustering:

$$\{\, p^{(k)}(s_j \mid s_i) \,\} \longrightarrow \{p^{(k)}(C_{q(s_j)} \mid C_{q(s_i)}),\ p^{(k)}(s_j \mid C_{q(s_j)}) \,\}$$

- **Step 2** Bi-multigram reestimation:

$$\{\, p^{(k)}(C_{q(s_j)} \mid C_{q(s_i)}),\ p^{(k)}(s_j \mid C_{q(s_j)}) \,\} \longrightarrow \{p^{(k+1)}(s_j \mid s_i) \,\}$$

Step 1 takes a phrase distribution as an input, assigns each phrase s_j to a class $C_{q(s_j)}$, and outputs the corresponding class distribution. In our experiments, the class assignment is performed by maximizing the mutual information between adjacent phrases, following the line described in (Brown

et al., 1992), with only the modification that candidates to clustering are phrases instead of words. The clustering process is initialized by assigning each phrase to its own class. The loss in average mutual information when merging 2 classes is computed for every pair of classes, and the 2 classes for which the loss is minimal are merged. After each merge, the loss values are updated and the process is repeated till the required number of classes is obtained.
Step 2 consists in reestimating a phrase distribution using the bi-multigram reestimation equation (4) or (5), with the only difference that the likelihood of a parse, instead of being computed as in Eq. (2), is now computed with the class estimates, i.e. as:

$$\mathcal{L}(W,S) = \prod_{r} p(C_{q(s_{(r)})} \mid C_{q(s_{(r-1)})})\, p(s_{(r)} \mid C_{q(s_{(r)})}) \quad (6)$$

This is equivalent to reestimating $p^{(k+1)}(s_j \mid s_i)$ from $p^{(k)}(C_{q(s_j)} \mid C_{q(s_i)}) \times p^{(k)}(s_j \mid C_{q(s_j)})$, instead of $p^{(k)}(s_j \mid s_i)$ as was the case in section 2.1.
Overall, step 1 ensures that the class assignment based on the mutual information criterion is optimal with respect to the current estimates of the phrase distribution and step 2 ensures that the phrase distribution optimizes the likelihood computed according to (6) with the current estimates of the ciass distribution. The training data are thus iteratively structured in a fully integrated way, at both a paradigmatic level (step 1) and a syntagmatic level (step 2).

2.3 Interpolation of stochastic class-phrase and phrase models

With a class model, the probabilities of 2 phrases belonging to the same class are distinguished only according to their unigram probability. As it is unlikely that this loss of precision be compensated by the improved robustness of the estimates of the class distribution, class based models can be expected to deteriorate the likelihood of not only train but also test data, with respect to non-class based models. However, the performance of non-class models can be enhanced by interpolating their estimates with the class estimates. We first recall the way linear interpolation is performed with conventional word ngram models, and then we extend it to the case of our stochastic phrase-based approach. Usually, linear interpolation weights are computed so as to maximize the likelihood of cross evaluation data (Jelinek and Mercer, 1980). Denoting by λ and $(1-\lambda)$ the interpolation weights, and by p_+ the interpolated estimate, it comes for a word bigram model:
$p_+(w_j \mid w_i) =$

$$\lambda\, p(w_j \mid w_i) + (1-\lambda)\, p(C_{q(w_j)} \mid C_{q(w_i)})\, p(w_j \mid C_{q(w_j)}) \quad (7)$$

with λ having been iteratively estimated on a cross evaluation corpus W_{cross} as:

$$\lambda^{(k+1)} = \frac{1}{T_{cross}} \sum_{i,j} c(w_i w_j)\, \frac{\lambda^{(k)}\, p(w_j \mid w_i)}{p_+^{(k)}(w_j \mid w_i)} \quad (8)$$

where T_{cross} is the number of words in W_{cross}, and $c(w_i w_j)$ the number of co-occurences of the words w_i and w_j in W_{cross}.
In the case of a stochastic phrase based model - where the segmentation into phrases is not known *a priori* - the above computation of the interpolation weights still applies, however, it has to be embedded in dynamic programming to solve the ambiguity on the segmentation:

$$\lambda^{(k+1)} = \frac{1}{c(S^{*(k)})} \sum_{i,j} c(s_i s_j \mid S^{*(k)})\, \frac{\lambda^{(k)}\, p(s_j \mid s_i)}{p_+^{(k)}(s_j \mid s_i)} \quad (9)$$

where $S^{*(k)}$ the most likely segmentation of W_{cross} given the current estimates $p_+^{(k)}(s_j \mid s_i)$ can be retrieved with a Viterbi algorithm, and where $c(S^{*(k)})$ is the number of sequences in the segmentation $S^{*(k)}$. A more accurate, but computationally more involved solution would be to compute $\lambda^{(k+1)}$ as the expectation of $\frac{1}{c(S)} \sum_{i,j} c(s_i s_j \mid S)\, \frac{\lambda^{(k)}\, p(s_j \mid s_i)}{p_+^{(k)}(s_j \mid s_i)}$ over the set of segmentations $\{S\}$ on W_{cross}, using for this purpose a forward-backward algorithm. However in the experiments reported in section 4, we use Eq (9) only.

3 Experiments with phrase based models

3.1 Protocol and database

Evaluation protocol A motivation to learn bigram dependencies between variable length phrases is to improve the predictive capability of conventional word bigram models, while keeping the number of parameters in the model lower than in the word trigram case. The predictive capability is usually evaluated with the perplexity measure:

$$PP = e^{-\frac{1}{T}\log \mathcal{L}(W)}$$

where T is the number of words in W. The lower PP is, the more accurate the prediction of the model is. In the case of a stochastic model, there are actually 2 perplexity values PP and PP^* computed respectively from $\sum_S \mathcal{L}(W,S)$ and $\mathcal{L}(W,S^*)$. The difference $PP^* - PP$ is always positive or zero, and measures the average degree of ambiguity on a parse S of W, or equivalently the loss in terms of prediction accuracy, when the sentence likelihood is approximated with the likelihood of the best parse, as is done in a speech recognizer.

In section 3.2, we first evaluate the loss ($PP^* - PP$) using the forward-backward estimation procedure, and then we study the influence of the estimation procedure itself, i.e. Eq. (4) or (5), in terms of perplexity and model size (number of distinct 2-uplets of phrases in the model). Finally, we compare these results with the ones obtained with conventional n-gram models (the model size is thus the number of distinct n-uplets of words observed), using for this purpose the CMU-Cambridge toolkit (Clarkson and Rosenfeld, 1997).

Training protocol Experiments are reported for phrases having at most n = 1, 2, 3 or 4 words (for n =1, bi-multigrams correspond to conventional bigrams). The bi-multigram probabilities are initialized using the relative frequencies of all the 2-uplets of phrases observed in the training corpus, and they are reestimated with 6 iterations. The dictionaries of phrases are pruned by discarding all phrases occuring less than 20 times at initialization, and less than 10 times after each iteration[2], except for the 1-word phrases which are kept with a number of occurrences set to 1. Besides, bi-multigram and n-gram probabilities are smoothed with the backoff smoothing technique (Katz, 1987) using Witten-Bell discounting (Witten and Bell, 1991)[3].

Database Experiments are run on ATR travel arrangement data (see Tab. 1). This database consists of semi-spontaneous dialogues between a hotel clerk and a customer asking for travel/accomodation informations. All hesitation words and false starts were mapped to a single marker "*uh*".

	Train	test
Nb sentences	13 650	2 430
Nb tokens	167 000	29 000 (1 % OOV)
Vocabulary	3 525	+ 280 OOV

Table 1: ATR Travel Arrangement Data

3.2 Results

Ambiguity on a parse (Table 2) The difference ($PP^* - PP$) usually remains within about 1 point of perplexity, meaning that the average ambiguity on a parse is low, so that relying on the single best parse should not decrease the accuracy of the prediction very much.

Influence of the estimation procedure (Table 3) As far as perplexity values are concerned, the estimation scheme seems to have very little influence, with only a slight advantage in using the forward-backward training. On the other hand, the size of the model at the end of the training is about 30% less with the forward-backward training: approximately 40 000 versus 60 000, for a same test perplexity value. The bi-multigram results tend to indicate that the pruning heuristic used to discard phrases does not allow us to fully avoid overtraining, since perplexities with n =3, 4 (i.e. dependencies possibly spanning over 6 or 8 words) are higher than with n =2 (dependencies limited to 4 words).

n	1	2	3	4
PP	56.0	43.9	44.2	45.0
PP^*	56.0	45.1	45.4	46.3

Table 2: Ambiguity on a parse.

Test perplexity values PP^*				
n	1	2	3	4
F.-B.	56.0	45.1	45.4	46.3
Viterbi	56.0	45.7	45.9	46.2
Model size				
n	1	2	3	4
F.-B.	32505	42347	43672	43186
Viterbi	32505	65141	67258	67295

Table 3: Influence of the estimation procedure: forward-backward (F.-B.) or Viterbi.

Comparison with n-grams (Table 4) The lowest bi-multigram perplexity (43.9) is still higher than the trigram score, but it is much closer to the trigram value (40.4) than to the bigram one (56.0) [4]. The number of entries in the bi-multigram model is much less than in the trigram model (45000 versus 75000), which illustrates the ability of the model to select most relevant phrases.

Test perplexity values PP				
n (and n)	1	2	3	4
n-gram	314.2	56.0	40.4	39.8
bimultigrams	56.0	43.9	44.2	45.0
Model size				
n (and n)	1	2	3	4
n-gram	3526	32505	75511	112148
bimultigrams	32505	42347	43672	43186

Table 4: Comparison with n-grams: Test perplexity values and model size.

[2] Using different pruning thresholds values did not dramatically affect the results on our data, provided that the threshold at initialization is in the range 20-40, and that the threshold of the iterations is less than 10.

[3] The Witten-Bell discounting was chosen, because it yielded the best perplexity scores with conventional n-grams on our test data.

[4] Besides, the trigram score depends on the discounted scheme: with a linear discounting, the trigram perplexity on our test data was 48.1.

4 Experiments with class-phrase based models

4.1 Protocol and database

Evaluation protocol In section 4.2, we compare class versions and interpolated versions of the bigram, trigram and bi-multigram models, in terms of perplexity values and of model size. For bigrams (resp. trigrams) of classes, the size of the model is the number of distinct 2-uplets (resp. 3-uplets) of word-classes observed, plus the size of the vocabulary. For the class version of the bi-multigrams, the size of the model is the number of distinct 2-uplets of phrase-classes, plus the number of distinct phrases maintained. In section 4.3, we show samples from classes of up to 5-word phrases, to illustrate the potential benefit of clustering relatively long and variable-length phrases for issues related to language understanding.

Training protocol All non-class models are the same as in section 3. The class-phrase models are trained with 5 iterations of the algorithm described in section 2.2: each iteration consists in clustering the phrases into 300 phrase-classes (step 1), and in reestimating the phrase distribution (step 2) with Eq. (4). The bigrams and trigrams of classes are estimated based on 300 word-classes derived with the same clustering algorithm as the one used to cluster the phrases. The estimates of all the class ditributions are smoothed with the backoff technique like in section 3. Linear interpolation weights between the class and non-class models are estimated based on Eq. (8) in the case of the bigram or trigram models, and on Eq.(9) in the case of the bi-multigram model.

Database The training and test data used to train and evaluate the models are the same as the ones described in Table 1. We use an additional set of 7350 sentences and 55000 word tokens to estimate the interpolation weights of the interpolated models.

4.2 Results

The perplexity scores obtained with the non-class, class and interpolated versions of a bi-multigram model (limiting to 2 words the size of a phrase), and of the bigram and trigram models are in Table 5. Linear interpolation with the class based models allows us to improve each model's performance by about 2 points of perplexity: the Viterbi perplexity score of the interpolated bi-multigrams (43.5) remains intermediate between the bigram (54.7) and trigram (38.6) scores. However in the trigram case, the enhancement of the performance is obtained at the expense of a great increase of the number of entries in the interpolated model (139256 entries). In the bi-multigram case, the augmentation of the model size is much less (63972 entries). As a result, the interpolated bi-multigram model still has fewer entries than the word based trigram model (75511 entries), while its Viterbi perplexity score comes even closer to the word trigram score (43.5 versus 40.4). Further experiments studying the influence of the threshold values and of the number of classes still need to be performed to optimize the performances for all models.

Test perplexity values PP^*			
	non-class	class	interpolated
bigrams	56.04	66.3	54.7
bimultigrams	45.1	57.4	43.5
trigrams	40.4	49.3	38.6
Model size			
	non-class	class	interpolated
bigrams	32505	20471	52976
bimultigrams	42347	21625	63972
trigrams	75511	63745	139256

Table 5: Comparison of class-phrase bi-multigrams and of class-word bigrams and trigrams: Test perplexity values and model size.

4.3 Examples

Clustering variable-length phrases may provide a natural way of dealing with some of the language disfluencies which characterize spontaneous utterances, like the insertion of hesitation words for instance. To illustrate this point, examples of phrases which were merged into a common cluster during the training of a model allowing phrases of up to $n = 5$ words are listed in Table 6 (the phrases containing the hesitation marker "*uh*" are in the upper part of the table). It is often the case that phrases differing mainly because of a speaker hesitation are merged together.

Table 6 also illustrates another motivation for phrase retrieval and clustering, apart from word prediction, which is to address issues related to topic identification, dialogue modeling and language understanding (Kawahara et al., 1997). Indeed, though the clustered phrases in our experiments were derived fully blindly, i.e. with no semantic/pragmatic information, intra-class phrases often display a strong semantic correlation. To make this approach effectively usable for speech understanding, constraints derived from semantic or pragmatic knowledge (like speech act tag of the utterance for instance) could be placed on the phrase clustering process.

5 Conclusion

An algorithm to derive variable-length phrases assuming bigram dependencies between the phrases has been proposed for a language modeling task. It has been shown how a paradigmatic element could

{ yes_that_will ; *uh*_that_would }
{ yes_that_will_be ; *uh*_yes_that's }
{ *uh*_by_the ; and_by_the }
{ yes_*uh*_i ; i_see_i }
{ okay_i_understand ; *uh*_yes_please }
{ could_you_recommend ; *uh*_is_there }
{ *uh*_could_you_tell ; and_could_you_tell }
{ so_that_will ; yes_that_will ; yes_that_would ; uh*_that_would }
{ if_possible_i'd_like ; we_would_like ; *uh*_i_want }
{ that_sounds_good ; *uh*_i_understand }
{ *uh*_i_really ; *uh*_i_don't }
{ *uh*_i'm_staying ; and_i'm_staying }
{ all_right_we ; *uh*_yes_i }

{ good_morning ; good_afternoon ; hello }
{ sorry_to_keep_you_waiting ; hello_front_desk ; thank_you_very_much ; thank_you_for_calling ; you're_very_welcome ; yes_that's_correct ; yes_that's_right }
{ non_smoking ; western_style ; first_class ; japanese_style }
{ familiar_with ; in_charge_of }
{ could_you_tell_me ; do_you_know }
{ how_long ; how_much ; what_time ; uh*_what_time ; *uh*_how_much ; and_how_much ; and_what_time }
{ explain ; tell_us ; tell_me ; tell_me_about ; tell_me_what ; tell_me_how ; tell_me_how_much ; tell_me_the ; give_me ; give_me_the ; give_me_your ; please_tell_me }
{ are_there ; are_there_any ; if_there_are ; if_there_is ; if_you_have ; if_there's ; do_you_have ; do_you_have_a ; do_you_have_any ; we_have_two ; is_there ; is_there_any ; is_there_a ; is_there_anything ; *uh*_is_there ; uh*_do_you_have }
{ tomorrow_morning ; nine_o'clock ; eight_o'clock ; seven_o'clock ; three_p.m. ; august_tenth ; in_the_morning ; six_p.m. ; six_o'clock }
{ we'd_like ; i'd_like ; i_would_like }
{ that'll_be_fine ; that's_fine ; i_understand }
{ kazuko_suzuki ; mary ; mary_phillips ; thomas_nelson ; suzuki ; amy_harris ; john ; john_phillips }
{ fine ; no_problem ; anything_else }
{ return_the_car ; pick_it_up }
{ todaiji ; kofukuji ; brooklyn ; enryakuji ; hiroshima ; las_vegas ; salt_lake_city ; chicago ; kinkakuji ; manhattan ; miami ; kyoto_station ; this_hotel ; our_hotel ; your_hotel ; the_airport ; the_hotel }

Table 6: Example of phrases assigned to a common cluster, with a model allowing up to 5-word phrases (clusters are delimited with curly brackets)

be integrated within this framework, allowing to assign common labels to phrases having a different length. Experiments on a task oriented corpus have shown that structuring sentences into phrases results in large reductions in the bigram perplexity value, while still keeping the number of entries in the language model much lower than in a trigram model, especially when these models are interpolated with class based models. These results might be further improved by finding a more efficient pruning strategy, allowing the learning of even longer dependencies without over-training, and by further experimenting with the class version of the phrase-based model.

Additionally, the semantic relevance of the clusters of phrases motivates the use of this approach in the areas of dialogue modeling and language understanding. In that case, semantic/pragmatic informations could be used to constrain the clustering of the phrases.

Appendix: Forward-backward algorithm for the estimation of the bi-multigram parameters

Equation (4) can be implemented at a complexity of $O(n^2T)$, with n the maximal length of a sequence and T the number of words in the corpus, using a forward-backward algorithm. Basically, it consists in re-arranging the order of the summations of the numerator and denominator of Eq. (4): the likelihood values of all the segmentations where sequence s_j occurs after sequence s_i, with sequence s_i ending at the word at rank (t), are summed up first; and then the summation is completed by summing over t. The cumulated likelihood of all the segmentations where s_j follows s_i, and s_i ends at (t), can be directly computed as a product of a forward and of a backward variable. The forward variable represents the likelihood of the first t words, where the last l_i words are constrained to form a sequence:

$$\alpha(t, l_i) = \mathcal{L}(W_{(1)}^{(t-l_i)} [W_{(t-l_i+1)}^{(t)}])$$

The backward variable represents the conditional likelihood of the last $(T-t)$ words, knowing that they are preceded by the sequence $[w_{(t-l_j+1)} \ldots w_{(t)}]$:

$$\beta(t, l_j) = \mathcal{L}(W_{(t+1)}^{(T)} \mid [W_{(t-l_j+1)}^{(t)}])$$

Assuming that the likelihood of a parse is computed according to Eq. (2), then the reestimation equation (4) can be rewritten as shown in Tab. 7.

The variables α and β can be calculated according to the following recursion equations (assuming a start and an end symbol at rank $t = 0$ and $t = T+1$):

$$p^{(k+1)}(s_j \mid s_i) = \frac{\sum_{t=1}^{T} \alpha(t,\ l_i)\ p^{(k)}(s_j \mid s_i)\ \beta(t+l_j,\ l_j)\ \delta_i(t-l_i+1)\ \delta_j(t+1)}{\sum_t \alpha(t,\ l_i)\ \beta(t,\ l_i)\ \delta_i(t-l_i+1)}$$

l_i and l_j refer respectively to the lengths of the sequences s_i and s_j, and where the Kronecker function $\delta_k(t)$ equals 1 if the word sequence starting at rank t is s_k, and equals 0 if not.

Table 7: Forward-backward reestimation

for $1 \leq t \leq T+1$, and $1 \leq l_i \leq n$:

$$\alpha(t, l_i) = \sum_{l=1}^{n} \alpha(t-l_i, l)\ p([W_{(t-l_i+1)}^{(t)}] \mid [W_{t-l_i-l+1}^{(t-l_i)}])$$

$$\alpha(0,1) = 1,\ \alpha(0,2) = \ldots = \alpha(0,n) = 0.$$

for $0 \leq t \leq T$, and $1 \leq l_j \leq n$:

$$\beta(t, l_j) = \sum_{l=1}^{n} p([W_{(t+1)}^{(t+l)}] \mid [W_{(t-l_j+1)}^{(t)}])\ \beta(t+l, l)$$

$$\beta(T+1,1) = 1,\ \beta(T+1,2) = \ldots = \beta(T+1,n) = 0.$$

In the case where the likelihood of a parse is computed with the class assumption, i.e. according to (6), the term $p^{(k)}(s_j \mid s_i)$ in the reestimation equation shown in Table 7 should be replaced by its class equivalent, i.e. by $p^{(k)}(C_{q(s_j)} \mid C_{q(s_i)})\ p^{(k)}(s_j \mid C_{q(s_j)})$. In the recursion equation of α, the term $p([W_{(t-l_i+1)}^{(t)}] \mid [W_{t-l_i-l+1}^{(t-l_i)}])$ is replaced by the corresponding class bigram probability multiplied by the class conditional probability of the sequence $[W_{(t-l_i+1)}^{(t)}]$. A similar change affects the recursion equation of β, with $p([W_{(t+1)}^{(t+l)}] \mid [W_{(t-l_j+1)}^{(t)}])$ being replaced by the corresponding class bigram probability multiplied by the class conditional probability of the sequence $[W_{(t+1)}^{(t+l)}]$.

References

F. Bimbot, R. Pieraccini, E. Levin, and B. Atal. 1995. Variable-length sequence modeling: Multigrams. *IEEE Signal Processing Letters*, 2(6), June.

P.F. Brown, V.J. Della Pietra, P.V. de Souza, J.C. Lai, and R.L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

P. Clarkson and R. Rosenfeld. 1997. Statistical language modeling using the cmu-cambridge toolkit. *Proceedings of EUROSPEECH 97.*

S. Deligne and F. Bimbot. 1995. Language modeling by variable length sequences: theoretical formulation and evaluation of multigrams. *Proceedings of ICASSP 95.*

S. Deligne, F. Yvon, and F. Bimbot. 1996. Introducing statistical dependencies and structural constraints in variable-length sequence models. In *Grammatical Inference : Learning Syntax from Sentences*, Lecture Notes in Artificial Intelligence 1147, pages 156–167. Springer.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistics Society*, 39(1):1–38.

F. Jelinek and R.L. Mercer. 1980. Interpolated estimation of markov source parameters from sparse data. *Proceedings of the workshop on Pattern Recognition in Practice*, pages 381–397.

S. M. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Trans. on Acoustic, Speech, and Signal Processing*, 35(3):400–401, March.

T. Kawahara, S. Doshita, and C. H. Lee. 1997. Phrase language models for detection and verification-based speech understanding. *Proceedings of the 1997 IEEE workshop on Automatic Speech Recognition and Understanding*, pages 49–56, December.

H. Masataki and Y. Sagisaka. 1996. Variable-order n-gram generation by word-class splitting and consecutive word grouping. *Proceedings of ICASSP 96.*

S. Matsunaga and S. Sagayama. 1997. Variable-length language modeling integrating global constraints. *Proceedings of EUROSPEECH 97.*

K. Ries, F. D. Buo, and A. Waibel. 1996. Class phrase models for language modeling. *Proceedings of ICSLP 96.*

M. Siu. 1998. *Learning local lexical structure in spontaneous speech language modeling.* Ph.D. thesis, Boston University.

B. Suhm and A. Waibel. 1994. Towards better language models for spontaneous speech. *Proceedings of ICSLP 94.*

I.H. Witten and T.C. Bell. 1991. The zero-frequency problem: estimating the probabilities of novel events in adaptative text compression. *IEEE Trans. on Information Theory*, 37(4):1085–1094, July.