TR-SLT-0046
Comparative Study on Multi-Class Composite N-grams Applied to English and Japanese

Fadi BADRA, Hirofumi YAMAMOTO

August, 27th 2003

Abstract

This document will present the research work I conducted in ATR for five months from april to august 2003. My work mainly consisted in an application of Multi-Class Composite N-gram language models, proposed recently by ATR researchers H.Yamamoto, S.Isogai and Y.Sagisaka (Yamamoto, 2001) and only applied to Japanese language so far, to the English language. The purpose of the experiments I conducted was to provide experimental data on such a model for the two languages, and determine to which extend this new technique, which showed good results for Japanese, can be applied as well to the English model. These tests were performed for training corpora of different sizes, extracted from the B.T.E.C., and running these experiments on the two languages in the same conditions enabled us to make a comparison between them. The results showed that Multi-Class language models improve conventional Class-Based ones for English too, but with different optimal connectivity information and only for small training corpora.

Table Of Contents

Acknowledgments

First of all, I would like to thank Dr.Kikui Genichiro (ATR-SLT Dept.2 head) for accepting me in this department and Dr.Yamamoto Hirofumi (ATR-SLT Researcher) for his precious help along this study.

But I would also like to thank very much all the dept 2 members for their warm welcome in ATR and in Japan.

Thank you so much to Makiko Tatsumi and Yukiko Ishikawa (ATR Staff Support Group), you had a lot of work with me...

As well, thank you to Laurent Besacier (CLIPS-France "maitre de conférences") for his support from France

Abstract

This document will present the research work I conducted in ATR for five months from april to august 2003. My work mainly consisted in an application of Multi-Class Composite N-gram language models, proposed recently by ATR researchers H.Yamamoto, S.Isogai and Y.Sagisaka (Yamamoto, 2001) and only applied to Japanese language so far, to the English language. The purpose of the experiments I conducted was to provide experimental data on such a model for the two languages, and determine to which extend this new technique, which showed good results for Japanese, can be applied as well to the English model. These tests were performed for training corpora of different sizes, extracted from the B.T.E.C., and running these experiments on the two languages in the same conditions enabled us to make a comparison between them. The results showed that Multi-Class language models improve conventional Class-Based ones for English too, but with different optimal connectivity information and only for small training corpora.

Introduction

In speech recognition, language models aim at validating acoustic hypothesis among sequences of words. They gather linguistic and stochastic information about a language, which are to be used in real-time decoding to predict words. Each word is assigned various probabilities of occurrence depending on its context, i.e. its preceeding and following words in the sentence. These models are trained on huge text corpora, and from this corpus analysis are computed the word probabilities to be used in the decoder.

The most widely used language models are word N-grams, in which only a limited context of N-1 words is taken into account to predict a word. But these models are based on a frequency analysis of a corpus, so they need a lot of training data to establish accurate probability values, and it deteriorates quickly when training corpora are small. Besides, the number of parameters increases exponentially with N. So increasing the N-gram order dramatically increases the quantities of training data required to produce accurate values.

To cope with this robustness problem in case of sparse training corpora, class N-grams have been introduced. The idea is to merge words into classes according to their common properties so that word prediction is replaced by class prediction. The great advantage of such a technique is that classes are much less numerous. So in addition to the gain of robustness, it can also highly reduce the size of the language model.

The language model proposed by ATR researchers (Yamamoto, 2001) is an extension of these class-based language models. In this model, frequent successions of words are grouped in a single lexicon entry to include higher order N-grams, and multi-class assignment is introduced to express that a word can have different right and left context dependance.

Background

The Multi-Class language model has already been tested on Japanese. It has had satisfying results, achieving 9.5% perplexity reduction compared to word N-grams and 16% Word Error Rate reduction. It also reduced the parameter size of 40%. It showed to be a great improvement of class-based models for limited training corpora, i.e. corpora of less than a million words. But no results were produced for English. So tests had to be run on the English corpus to decide to which extend this model can be applied to English as well. A similar study is being made as well by researchers on the Chinese language, in order to set a comparison between the 3 languages. Besides, in the first months of this study, a parallel research was lead to improve the tagging scheme of the current English corpus used by ATR.

Objectives

When we started these experiments, our first concern was to produce significant experimental data on the Multi-Class Composite Bigram language model applied to English. The parameters used to build such a language model can be assign different values. The language model can be trained on corpora of various sizes. The number of classes the words are merged into using ATR clustering technique can vary as well. And in the Multi-Class approach, either right or left markovian context dependance, or even a weighted combination of the two can be taken into account when merging words into classes. The purpose of this experiment was thus first provide an evaluation of the English language model for different values of these parameters : Corpus Size, Number of Classes and Connectivity Information.

Then our second objective was to make a comparison between English and Japanese. This is achieved by making the same experiment on the corresponding Japanese corpus, and in the same experimental conditions.

Finally, we attempt to decide, from the analysis of the results, to which extend the Multi-Class Composite language model is an appropriate model for the English language.

Language Modeling

As the grammar point of view has largely given way to the stochastic one in recent research activities, we will talk of a **language model** as the **set of probabilities of all word sequences of a language**. LANGUAGE MODEL

So a language model is built for each natural language, and aims at validating acoustic hypothese. For example, the language model will be referred to to choose from the close acoustic sentences “we won’t some money”, “we went some money” and “we want some money”, pointing out that the last one is more likely to have been spoken.

Statistical processing of natural language is based on the analysis of huge text **corpora**. (for example a collection of the last 20 years of Le Monde newspaper for French language, or the Brown corpus for English language, which is a collection of 500 english texts from different genres). Word sequences probabilities are thus estimated from their frequencies of occurrence in a chosen corpus. These values are computed once, during the corpus training phase, and since then kept in memory, in order to be used by the real-time decoding machine during speech recognition. CORPUS

Ideally, a language model should provide any probability of word sequences for any words of the natural language, including proper nouns, abbreviations, onomatopes, etc., i.e. all the words that could be spoken to the speech recognition system, and this would be obtained from the analysis of an infinite corpus. In practice, a limited number of words are given information on by the language model. They are gathered in a lexicon. A **lexicon** is a repository of words. It contains the vocabulary words, to which is usually added some information on them. like tags, pronounciation, definitions, etc. So the size of a language model is directly linked to the lexicon size. If other words are spoken, their probabilities won’t be evaluated during the training phase and an arbitrary value will generally be attributed to them. LEXICON

The language model should give reliable information for a lexicon word whatever its context is. The probability of a word sequence $w_1 w_2 ... w_n$ is computed as :

$$p(w_1 w_2 ... w_n) = p(w_1).p(w_2|w_1)...p(w_n|wi, i < n),$$

For example,
p(Today some teachers are on strike)=
p(Today).p(some | Today).p(teachers | Today some).p(are | Today, some teachers).
p(on | Today some teachers are).p(strike | Today some teachers are on).

As a result, a language model should give for any lexicon word its probability of appearance, given its whole history, i.e. for $w_n, p(w_n|w_i, i < n)$. Of course, such a model is impossible to produce fully, because it requires an infinite set of probabilities. As a result, it needs to be approximated.

N-grams

N-GRAM

One of the most common approximation for a language model is the **N-gram model**. The idea is to restrict the history of a word to its N-1 preceeding ones when predicting it. For example, in a bigram model, each word will be predicted using only the word preceeding it : p(strike | Today some teachers are on) is approximated by p(strike | on). In practice, only 2, 3 or 4-grams are used. ATR researchers are mainly using bigrams for their experiments.

MAXIMUM LIKELIHOOD ESTIMATION

The conditional probabilities are estimated by their frequency of appearance, computed on a large corpus. It is the **Maximum Likelihood Estimation** (M.L.E.):

$$p_{MLE}(w_n|w_{n-2}w_{n-1}) = \frac{N(w_{n-2}w_{n-1}w_n)}{N(w_{n-2}w_{n-1})}$$

where $N(w_{n-2}w_{n-1})$ is the number of occurrences of the sequence $w_{n-2}w_{n-1}$ in the corpus.

In the N-gram approximation, a conditional probability $p(w_i|w_{i-N+1}, ...w_{i-1})$ has to be produced for every succession of N lexicon words $w_{i-N+1}, ...w_{i-1}, w_i$. If V denotes the size of the lexion, V^N probability values have to be computed in such a model. And they are to be kept in the decoder's memory, in order to be used in real-time speech recognition to evaluate the probability of a succession of words.

Smoothing

In language modeling, probabilities of word sequences are obtained from the analysis of a limited corpus. As a result, word sequences, which do not occur in the training corpus won't have their probability evaluated during the training phase. That is, the n-gram matrix for any given training corpus is **sparse**. However, speech recognition will have to be able to evaluate the probability of a vocabulary word sequence which did not occur in the training corpus. Therefore, the probability of such word sequences should not be zero, i.e. $\forall i, p(w_i|w_{i-2}, w_{i-2}) \neq 0$

SMOOTHING

The main purpose of **smoothing** is to assign non-zero probabilities to these vocabulary word sequences which were not evaluated in the corpus analysis.

Clustering

To provide reliable information, the N-gram probabilities must be assessed on a large corpus. Indeed, the larger the corpus is, the more accurate the probabilities will be, since they are computed from a frequency estimation (the M.L.E.). So for a specific vocabulary, a large corpus including occurrences of its word sequences must be gathered. And as a consequence, the higher the order N is, the larger the training corpus must be to provide accurate probability values.
Furthermore, the vocabulary itself has to be large enough to leave as few out-of-vocabulary words as possible in real-time decoding. But as we saw increasing the vocabulary has a direct impact on the number of probabilities to compute, since the number of probabilities increases exponentially with the vocabulary size - for a vocabulary size of V, a N-gram model would need V^N probabilities. As a result, this may introduce a problem of memory size in the real-time decoding machine.

CLASS-BASED MODELS

To face these two issues - corpus sparseness and memory size, words are often put into classes, which are determined according to their grammatical or semantic behaviors. Then the prediction of words will be changed into predictions of classes. The bigram probability $p(y|x)$ is then approximated by the probability $p(C(y)|C(x)).p(y|C(y))$.

Using class-based models, we gain in robustness. Indeed, for small training corpora, the language model can then be enhanced because it has the effect of a lexicon size reduction; that is the classes are much less numerous than the words, so the frequency estimations on classes would be much more accurate. And some information about a word sequence can even be learnt through its class. For example, if the numbers are gathered in the same classe NUMBER, for they have the same behavior in a sentence, and if the number 9 does not appear in the corpus, its probability can be assessed anyway thanks to its class NUMBER. Then p(nine years old) will be computed using p(NUMBER years old). But of course, if the corpus is large, the class-based approximation can induce a loss of information compared to the corresponding word-based model.
In addition, this results in a substantial reduction of the language model parameters. Actually, if C denotes the number of classes the words are merged in, the number of parameters to be kept in memory would be C^N + V instead of V^N.

Different Types of Clustering

HARD/SOFT CLUSTERING

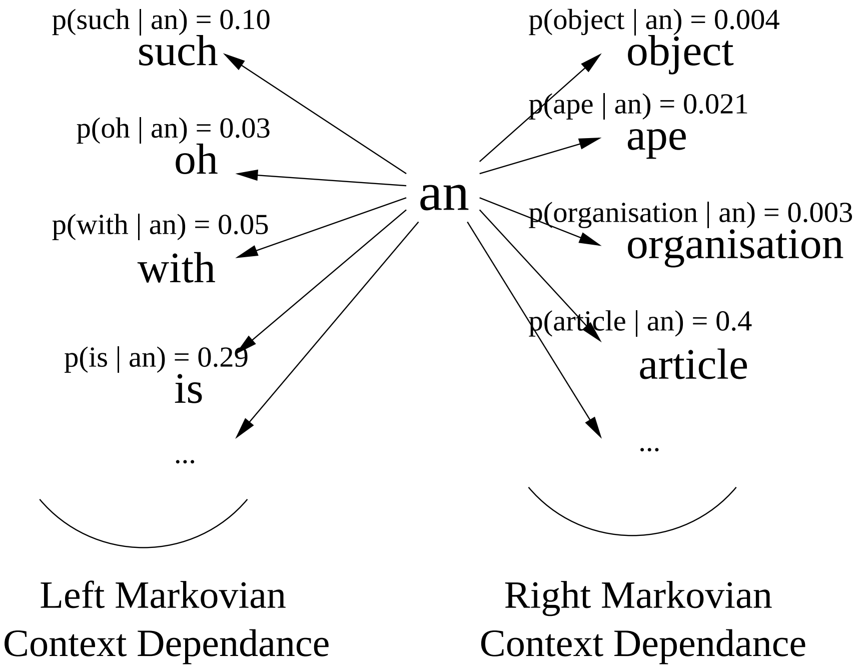
When defining classes, we can decide that a single word will belong to only one class (**hard clustering**), or that it can belong to many of them (**soft clustering**). Besides, the choice of classes can be **knowledge-based** or **data-driven**. In knowledge-based clustering, classes are defined according to syntactic or semantic information, and they are fixed before any word is put in it. The most common knowledge-based clustering is probably the Part-Of-Speech one , which clusters the

PART-OF-SPEECH

words according to their grammatical role, i.e. making a VERB class, a NOUN class,etc. The P.O.S clustering used for english in ATR is detailed in annexe A. In data-driven clustering, words are usually clustered automatically in such a way that the lost information is minimal. The indicator used is often perplexity.

ATR Clustering method

In ATR, the first clustering technique performed when building the english language model is Part-Of-Speech clustering. The words are put into different clusters according to their tags. Then, inside these classes is performed another clustering, which is then data-driven.
The data driven clustering is, as detailed in (Yamamoto, 2001), performed in order to minimize the loss of information in terms of markovian context dependances. To define the markovian context dependances, let's consider a word w, and place ourselves in the case of a bigram language model - this can easily be generalized to higher order N-grams. In a word sequence, this word can be followed by any lexicon word. And the word based language model provides bigram probabilities $p(x|w)$ of any lexicon word x succeeding w :



Then we would talk of right markovian context dependance of a word w to denote the vector of probabilities $p(x|w)$ for each lexicon word x. A left context dependance can also be define as the set of bigram probabilities $p(x|w)$ for any word x preceeding w. In data-driven clustering, two bigram probability vectors are first set for each word, one for each context dependance. One of them, which we'll call v^t represents

MARKOVIAN CONTEXT DEPENDANCE

its preceeding word connectivity and the other, which we'll call v^f represents its following word connectivity. For example, for the word w, the two vectors

$$v^t(w) = [p^t(w_1|w), p^t(w_2|w), ...p^t(w_N|w)]$$

$$v^f(w) = [p^f(w_1|w), p^f(w_2|w), ...p^f(w_N|w)]$$

are set. (N is there the size of the vocabulary) Then, to compute the clustering algorithm, a third vector composed of any of these vectors or both of them is used : v=$[v^t]$, $[v^f]$ or $[v^t, v^f]$. Starting with singleton classes, the clustering algorithm merges 2 classes at each iteration and this vector is recomputed. When two classes are merged, the right and left context dependance vectors can be recomputed using the class based approximation, but with the new set of classes : $p(y|x) \approx p(C(y)|C(x)).p(y|C(y))$ Then the choice of the two classes to be merged is the one for which the words merged have the less change among their context dependancies. The indicator used is an Euclidian distance between the current vector and the new vector obtained after the merge.

MULTI-CLASS CLUSTERING

In conventional class based language models, right and left markovian context dependance are equally taken into account when merging words into classes. Instead, the idea of **multi-class clustering** is to take into account only the right or the left context dependance of words, or a weighted combination of them. It models that words may have the same right or left context, but a different for the other side. Take 'a' and 'an', for example, or 'is' and 'are'. 'is' and 'are' have the same right context dependance, but they may not be put into the same class because their left context dependance is much different. Therefore, in the clustering algorithm, we would use a combination of right and left context dependance vectors v^t and v^f : v = $[\lambda v^t, (1 - \lambda)v^f]$ with $\lambda \in [0, 1]$. We can then build sets of classes using more forward information than backward, or vice versa. This technique helps particularly for languages with inflexion, such as French or Japanese.

Composite Words

COMPOSITE WORDS

Furthermore, another language model improvement used at ATR is the introduction of **Composite Words** in the lexicon. It is based on the observation that some sequences of words occur very frequently in the corpus. For example, the sequences "come on" or "a little bit of" are very common in english. So the idea is to consider these sequences as vocabulary words themselves. The most frequent word sequences in the corpus, that is the sequences which frequency is upon a fixed threshold, are changed to their corresponding composite word, and the latter is inserted in the lexicon as a new vocabulary word. For a word sequence A B C is created a composite word A+B+C, which resulting to-class is the word A's to-class, and which from-class is word C's from-class. The main advantage of such an operation is to be able to treat higher order N-grams as bigrams or tri-grams. For example, the 5-gram probability p(money|a little bit of) will be treated as a bigram probability. But as new lexicon entries are formed, the size of the language model increases, since N-gram probabilities of new sequences of words - using these new lexicon entries - have to be computed. And this could result in a loss of accuracy among the probabilities, especially if the training corpus is small. So the number of composite words to add to the vocabulary must be well chosen in order to improve the language model.

Performance Evaluation

Entropy and Perplexity are the most common metrics used to evaluate a language model.

Entropy is a measure of information, and its computation requires the establishement of a variable X which ranges over whatever we are predicting (words, letters, POS...the set of which we'll call χ) and that has a probability function p(x). The entropy of X is then : ENTROPY

$$H(X) = -\sum_{x \in \chi} p(x) log_2 p(x)$$

As we use here log base 2, the entropy will be measured in bits. Intuitively, entropy can be seen as a lower bound for the number of bits necessary to encode a certain peace of information. But this metric is easier to apprehend through its correlated metric called **Perplexity**, which value is 2^H. Perplexity, for word predictions, can be seen as the weighted average number of words we will have to choose from. So of course, the lower the perplexity is, the better. As the purpose of a language model is to help us choose between words for a given history, it should tend to having for each case the right word with a probability higher than the others. PERPLEXITY

To illustrate this idea, let's introduce another expression of entropy. For a given corpus $W = w_1 w_2 ... w_N$, the total entropy can be written :

$$H(W) = -log[p(w_1).p(w_2|w_1).p(w_3|w_1 w_2)...p(w_N|w_1 w_2 ... w_{N-1})]$$

In this expression, we can make some observations :

1. Entropy **strongly depends on the corpus**. And as a result, two language models can only be compared on the same corpus.

2. Entropy also depends on the definition of the variable measured. In our experiments, as a word can have many entries in the lexicon (one for each of its Part-Of-Speech tags), the entropy measured is the entropy of **tagged words**, i.e p(book [NOUN] | history) ≠ p(book [V] | history).

3. A good language model should maximize the probabilities $p(w_i|history)$ for each word w_i of the corpus.

Experimental Conditions

Vocabulary and Corpus

To run the experiments, we use ATR's **B**asic **T**ravel **E**xpressions **C**orpus. (Takesawa,2002) It is a bilingual English / Japanese broad-coverage corpus made of common japanese expressions and their translations extracted from phrase-books for foreign tourists. It contains about 200,000 sentences, which provide a 2.4 million words for the English language and 1.1 million for the Japanese one. This corpus covers the most common expressions of the daily life, but as they are read sentences they differ from spontaneous speech because they do not contain any idiomatic or argotic expression, nor any hesitation and are well articulated. Besides, the english sentences are only translations so it makes it even further from spontaneous speech. Anyway, for our experiments, this approximation will be enough since we are not looking for very accurate result values, but mainly looking for a global trend.

The lexicon is extracted from this corpus, and the most frequent sequences of words are grouped into composite words and added to it. (Yamamoto,2001) It contains around 30,000 words for the whole English B.T.E.C.. Any phonetic information has been removed from it and only words and their corresponding tags were kept.
Here's how the lexicon look like :

abbey$|||CN||$
abc_airlines$|||PROPN||$
absorb$||3S|V||$
abstract$|||ADJ||$
accidentally$|||ADV||$
etc.

From this lexicon were kept only the words having more than 2 occurrences in the corpus. This operation was performed in order to reduce the size of the lexicon, and as a result the running time of the clustering algorithm, so the total running time of the experiment becomes acceptable for a large number of tests. (the running time of the clustering algorithm is thus reduced from half a day to less than one hour). The remaining lexicon is usually cut by half.

The experiments were run on corpora of various sizes extracted from the BTEC corpus. A first experiment was made on the whole BTEC. Then the same experiment was run on fractions of it. Sub-corpora were extracted from the corpus taking successively one sentence every 2 sentences, one sentence every 4 sentences, then one every 8 and one every 16. These corpora were tagged with ATR POS tags and the entropy measured was the entropy of tagged words (using the formula $-\frac{1}{N}\sum_i log_2 p_i$, where p_i is the probability of the tagged bigram).

Tags

The English P.O.S tagset used for these experiments was ATR tagset as it was in April 2003. It contains 104 tags (listed in the annexe p.22), among which around 20 corresponded to no corpus word. Concerning the Japanese tagset, only the main tag (V, CN,...) were considered, plus the inflexion form tag, which makes 69 tags, but only when building the set of classes to be used for w_i, that is the word to predict in a bigram $p(w_i|w_{i-1})$. When building the set of classes for w_{i-1}, this tag was discarded. The reason for this is that for the Japanese language, the inflexion of a word only depends on the word following it, and in ATR multi-class language model applied to Japanese, the best results are obtained using only backward information for w_i. Without considering the inflexion form, the number of tags is 38.

Multi-Class Parameters

To build multi-class language models, bigram probabilities are approximated by corresponding class-based ones : $p(w_i|w_{i-1}) \approx p(C(w_i)|C(w_{i-1})).p(w|C(w_i))$. But to compute these probabilities, we need not to use the same set of classes for w_i and w_{i-1}. Therefore, two sets of classes were created. One that would be used for w_i and the other for w_{i-1} Then class-based bigram probabilities were computed from an analysis of the corpus. In this analysis, the frequencies were evaluated looking for each word of the bigram at which class it belongs. We do not even need to use the same number of classes for the two words - though we will as an approximation, assuming that it has a little influence on the results. Different connectivity information can also be used to build each set of classes.

From the results of preliminary tests using various configuration of forward and backward connectivity information to build each of the two sets of classes, we chose to use opposite connectivity information for each of the 2 sets of classes. For the one used for w_i, we would choose to make left (backward) context dependance count for λ of the total context dependance ($\lambda \in [0, 1]$), and to make right (forward) context dependance count for $(1 - \lambda)$. And for the one used for w_{i-1}, we would do the opposite. (i.e. weigh forward information by λ and backward by $(1 - \lambda)$.

Evaluation of the Language Models

The language models issued and evaluated in these experiments were only bigram ones smoothed with Good-Turing smoothing technique. (see annexe p.22) The language model evaluation did not include speech recognition tests. No Word Error Rate was produced. The models were only evaluated in terms of perplexity, calculated on the BTEC. The testsets used for these evaluations were corpora distinct from the training one. They were composed of 82k words for English and 71k words for Japanese.

Results

We produced several graphs (gathered as annexes to the present paper) representing for a given corpus size the perplexity results according to the number of classes used to build the sets of classes. The curves were also drawn for different connectivity information each time. We drew a curve for each value of λ, from 1.0 downto 0.0 by steps of 0.2, plus a curve for λ=0.5, which corresponds to the conventional (non multi-class) class-based language model. In these graphs, "1.0 b + 0.0 f / 0.0 b + 1.0 f" ($\lambda = 1.0$) means : only **b**ackward information was used to build the set of classes used for w_i, and only **f**orward information was used to build the set of classes used for w_{i-1}.

English Results

From the results obtained when applying Multi-Class Composite language models to English, we can say that :

- Multi-Class Composite language models applied to English show to be an improvement to the conventional Class-Based language models.
- Applying these models to English can lead to a gain of perplexity
- The optimal value of λ is an intermediate one, which is around 0.8.

But these conclusions are true only for **small training corpora**. Indeed, the multi-class model did not provide any improvement compared to conventional class-based model for training corpora which exceeded 200k words.

Comparison English / Japanese

From the comparison of the experimental data that we obtained for each of the two languages, a couple remarks can be made :

- First, the perplexity gain of the multi-class model compared to word based ones was always much lower for English than for Japanese, except for small training corpora. Figure 1 represents the maximal perplexity gains that were obtained for each language compared to the Word-Based perplexity and according to the training corpus size.

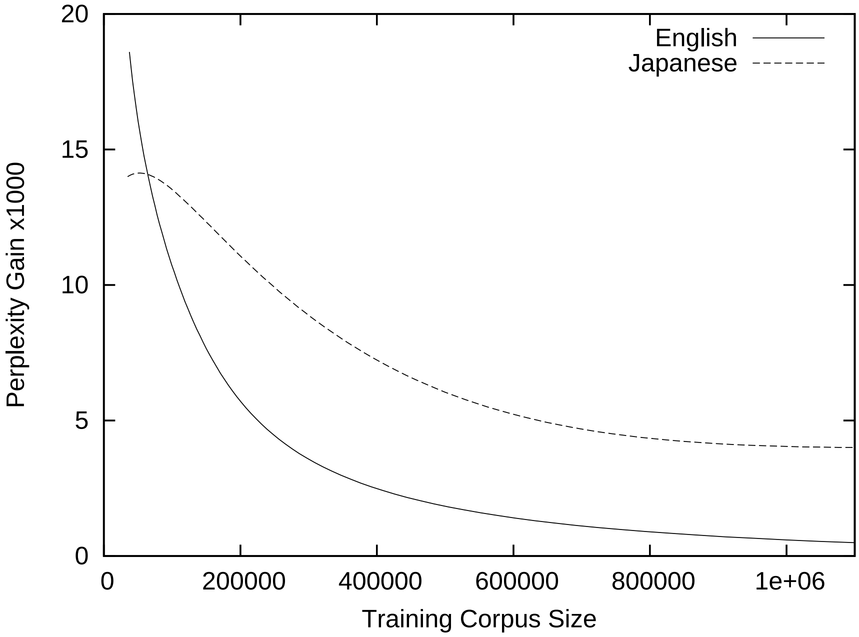


Figure 1: Perplexity Reduction achieved with Multi-Class Models

- The parameter size reduction that can be achieved by Multi-Class language model compared to word based one is lower for English, except for small training corpora. Figure 2 shows the Multi-Class parameter size for each language, as a percentage of the corresponding Word-Based one, according to the training corpus size.

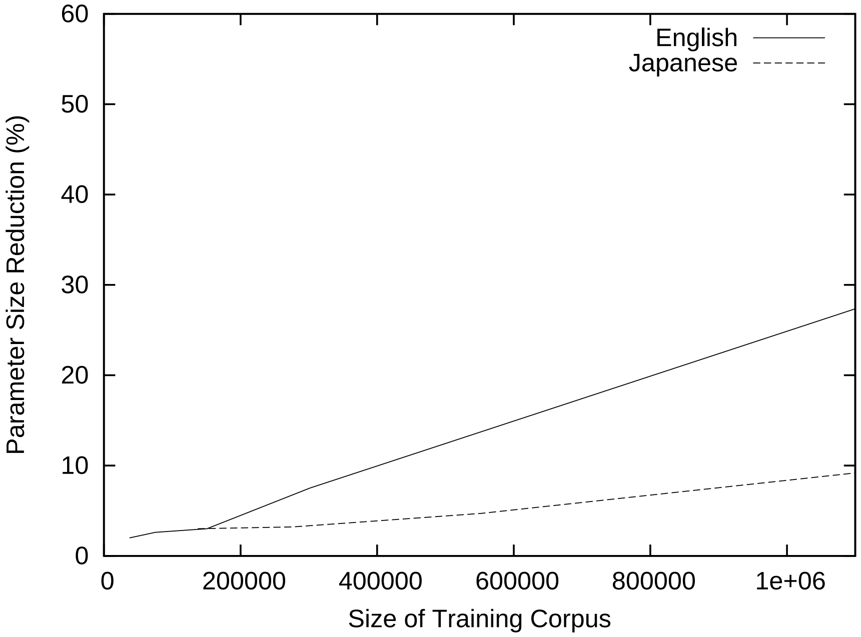


Figure 2: Parameter Size Reduction achieved with Multi-Class Language Models

- The optimal value of λ differs from a language to another, but this value does not depend on the training corpus size. The best value is around 0.8 for English, whereas Japanese models have their best results for λ=1.0.

Conclusion

In these experiments, we have applied Multi-Class Composite language model to ATR's English and Japanese BTEC. We ran the same experiment using various training corpus sizes, number of classes to merge the words in and using weighted combinations of right and left markovian dependances to build these classes. The results showed that this language model improves the conventional class based one for English too, increasing the perplexity gain and allowing a greater parameter size reduction. However, different context information has to be taken into account to build the Multi-Class language model. The optimal results were obtained for a value of λ around 0.8 for English whereas Japanese had best results for λ=1.0. But unlike the Japanese one, this improvement was only observed for small training corpora (less than 200,000 words). Actually it seems that the clustering technique itself is less efficient for English than for Japanese. This is probably due to the grammatical constraints of the Japanese language and the use of function words.

References

- **Books :**

  Daniel Jurafsky and James H. Martin, 2000, Speech and Language Processing, *Prentice Hall.*

  Manning and Shutze, 1999, Foundations of Statistical Natural Language Processing, *MIT press* .

- **Articles :**

  P.F.Brown, P.V. de Souza, R.L Mercer, V.J. della Pietra, J.C.Lai, Class-based n-gram Models of Natural Language.

  Jianfeng Gao, Joshua T. Goodman, Jiangbo Miao, 2001, The Use of Clustering Techniques for Language Modeling - Application to Asian Language. *Computational Linguistics and Chinese Language Processing*, Vol 6, No 1, pp 1-34.

  Joshua T. Goodman, Putting it all together : Language Model Combination.

  Joshua T. Goodman, 1999, An Empirical Study on Smoothing Techniques for Language Modeling, *Computer Speech and Language*, vol 13, nb 4.

  R.Kneser and H.Ney, Improved Backing-off For n-gram Language Modeling.

  P.Placeway, R.Schwartz, P.Fung and L.Nguyen, The Estimation of Powerful Language Models from Small and Large Corpora.

  T. Takezawa et al. (2002). Toward a Broad-coverage Bilingual Corpus for Speech Translation of Travel Conversations in the Real World. *Proc LREC*

  H. Yamamoto, S. Isogai and Y. Sagisaka, 2001, Multi-class Composite n-gram Language Model for Spoken Language Processing using Multiple Word Clusters.

H. Yamamoto and Y. Sagisaka, 1999, Multi-class composite n-gram based on connection direction. *Proc. ICASSP*, pages 533-536.

Annexes

Good-Turing Smoothing

Considering that the summed probabilities of n-grams must always be 1, i.e. for bigrams :

$$\sum_{w_i \in V} p(w_i|w_{i-1}) = 1,$$

the mass of probabilities assigned to out-of-corpus sequences will come from the non-zero probabilities. Actually, smoothing consist in **discounting** some non-zero counts in the corpus in order to get the probability mass that will be allowed to the zero counts. For a given history w_{i-1}, the total quantity that is taken from the non-zero probabilities $p(w_i|w_{i-1})$ is generally called β

Like the absolute discounting, the Good-Turing discounting smoothes counts of N-grams. The discounted number d depends on the count c of the word sequence. The basic idea is that high counts N-grams are less numerous in the corpus than low counts n-grams. So, assuming N_c to be the number of n-grams of count c in the corpus, the counts of the low count N-grams are multiplied by the factor $\frac{N_{c+1}}{N_c}$. However, this operation is made only for the lowest count N-grams. A maximum count k is chosen above which no operation is made on the N-grams. k is typically around 5.

English POS tags

Tag	Part Of Speech	Examples
CN	common nouns	bee, microphone, parrot
PROPN	proper noun	akira, chardonnay, kansai, sidney
PRON	pronoun	something, quite a few, these, it
POSS	possessive pronoun	hers, his, mine, ours, yours
NUM	numeral	zero, ninth, twelve
LETTER	graph	a, ab, o, ra, url, vt
V	verbs	add, belong, hike, paint

Tag	Part Of Speech	Examples
BEV	be verb	be, was, were
AUXV	auxilliary verb	can, do, may, will
HAVEAUXV	have auxilliary	have, had, had_better
DET	determiners and possessive pronouns	a, the, our, their
DETADJ	adjectival phrase It can be followed by a determiner.	almost, either, lot_of
PREP	prepositions	at, by, despite, of, until
ADJ	adjectives	brief, creative, fragile, glorious
ADV	adverbs	calmly, late, orally, roughly
PREADV	adverb which modifies verb, adjective and adverb	less, somewhat, quite, any
PREPADV	prepositional word, but noun (phrases) do not follow	across, on, over
LOCADV	locative adverbs	abroad, here, over_there
CONJADV	morpheme which functions as conjunction/adverb	however,though, otherwise
PRONADV	morpheme which functions as pronoun/adverb	elsewhere, little, well, much
WHADJ	interrogative which modifies following nouns	how many, how much
WHPRON	relative pronoun	what, which, whatever
WHADV	relative adverb	when, where, why
WHCONJ	conjonction which starts which interrogative	when, whenever
HOWADV	how adverb which modifies following adjective/verb	how, how much, how many
PREP	preposition	
CONJ	coordinative conjunction	and, but, or, since, whereas
INTERJ	interjections	bye_bye, ha, bingo, yeah
PRENOM	morpheme located right before proper nouns	
PNOM	morpheme which always come after nouns/numerals	ago, and_so_on, oćlock, san
NOT	word not	not
NO	word no	no
$S	the ś class	ś
VTO	verb + to	wanna, gotta
ADJTO	adjective + to	gonna, willing to
UH	interjection	ah,uh

Several tags are added to the former ones to precise its inflexion :

Tag	Part Of Speech	Examples
INGG	gerond	
INGP	progressive	
PAST	preterit form of a verb	became, didn't, broke_up, won
PP	past participle	assumed, trusted, undone, brought_back
ER	comparative	firmer, better
EST	superlative	best, latest, much