17

Linearly Interpolated Hierarchical N-gram Language Models for Speech Recognition Engines

Imed Zitouni[1] and Qiru Zhou[2]
[1]IBM T.J. Watson Research Center, NY, [2]Bell Labs Alcatel-Lucent, NJ USA

1. Introduction

Language modeling is a crucial component in natural language continuous speech recognition, due to the difficulty involved by continuous speech [1], [2]. Language modeling attempts to capture regularities in natural language for the purpose of improving the recognition performance. Many studies have shown that the word error rate of automatic speech recognition (ASR) systems decreases significantly when using statistical language models [3], [4], [5]. The purpose of language models (LMs) is to compute the probability $P(w_1^I)$ of a sequence of words $w_1^I = w_1, \ldots w_I$. The probability $P(w_1^I)$ can be expressed as: $P(w_1^I) = \prod_{i=1}^{I} P(w_i|h_i)$, where $h_i = w_1, \ldots, w_{i-1}$ is the history or the context of word w_i. The probability $P(w_1^I)$ becomes difficult to estimate as the number of words in h_i increases. To overcome this problem, we can introduce equivalent classes on the histories h_i in order to reduce their cardinality. The *n*-gram language models approximate the dependence of each word (regardless of *i*) to the $n-1$ words preceding it: $h_i \approx w_{i-n+1} \ldots w_{i-1}$. The probability $P(w_1^I)$ can then be expressed as:

$$P(w_1^I) = \prod_{i=1}^{I} P(w_i|w_{i-n+1} \ldots w_{i-1})$$

The *n*-gram approximation is based on the formal assumption that language is generated by a time-invariant Markov process [6].

Word *n*-gram LMs (mainly 2-gram and 3-gram LMs) are the most commonly used approach in language modeling. When enough data is available, word *n*-gram LMs have proved extremely useful in estimating the likelihood of frequently occurring *n*-grams, *($w_1, \ldots w_n$)*. When using this approach, estimating the probability of low-frequency and unseen *n*-grams is still inherently difficult. The problem becomes more acute as the vocabulary size increases since the number of low-frequency and unseen *n*-grams events increases considerably.

Automatic continuous speech recognition systems still make errors especially on unseen and rare events. Because of the Zipf's law [7], we will always expect unseen and rare events during recognition. Hence, the data sparseness constitutes a crucial point to take into account when building language models.

Source: Robust Speech Recognition and Understanding, Book edited by: Michael Grimm and Kristian Kroschel,
ISBN 987-3-90213-08-0, pp.460, I-Tech, Vienna, Austria, June 2007

Many approaches have been reported to overcome the probability estimation problem of low-frequency *n*-grams. One of them is the class *n*-gram language models [1], [8]. Using this approach, words are partitioned into equivalence classes, and the inter-word transition probability is assumed to depend only on the word classes. Class *n*-gram language models are more compact and generalize better on unseen *n*-grams compared to standard word-based language models. Nevertheless, for large training corpora, word *n*-gram language models are still better than class-based language models in capturing collocational relations between words.

A better approach is to build a language model that is general enough to better model unseen events, but specific enough to capture the ambiguous nature of words. Our solution is to hierarchically cluster the vocabulary words, building a word tree. The leaves represent individual words, while the nodes define clusters, or word classes: a node contains all the words of its descendant nodes. The closer a node is to the leaves, the more specific the corresponding class is. At the top of the tree, the root cluster contains all the words in the vocabulary. The tree is used to balance generalization ability and word specificity when estimating the probability of *n*-gram events. Then, we build a hierarchical *n*-gram language models that take benefit of the different information in every node of the tree to estimate the probability $P(w_i|w_{i-n+1}\dots w_{i-1})$ of a word w_i given its context $w_{i-n+1}\dots w_{i-1}$. This approach allows us to take advantage of both the power of word *n*-grams for frequent events and the predictive power of class *n*-grams for unseen or rare events.

One way to benefit from the word class hierarchy is to use the backoff hierarchical class *n*-gram language models (HCLMs) that we introduced recently [9], [10], [11], [12]. The backoff hierarchical class *n*-gram language models estimate the probability of an unseen event using the most specific class of the tree that guarantees a minimum number of occurrences of this event, hence allowing accurate estimation of the probability. This approach is a generalization of the well known backoff word *n*-gram language modeling technique [13]. The backoff word *n*-gram language modeling technique estimates the probability of an unseen *n*-gram (w_{i-n+1}^{i}) using a more general context, which is the (n-l)-gram (w_{i-n+2}^{i}). However, when using the backoff hierarchical class *n*-gram language models, the probability of an unseen *n*-gram (w_{i-n+1}^{i}) is computed according to a more specific context than the (n-l)-gram: we use the class of the most distant word w_{i-n+1} followed by the other words: $F(w_{i-n+1}), w_{i-n+2}^{i-1}$. The function *F(x)* represents the class (parent) of *x* within the hierarchical word tree, where *x* can be a class itself, or a single word, depending on its location in the class hierarchy.

In this chapter we introduce a novel language modeling technique named linearly interpolated hierarchical *n*-gram language models. This approach combine the power of word *n*-grams for frequent events and the predictive power of class *n*-grams for unseen or rare events. It linearly interpolate different *n*-gram LMs each one of them is trained on one level of the class hierarchy. The model trained on the leaves level (level 0) is the standard word *n*-gram language models. Those language models trained on a level in the class hierarchy greater than 0 are in fact the class *n*-gram language models. The higher the number of levels in the class hierarchy is, the more compact and general the class *n*-gram language models become.

In the next section we briefly describe previously published related works. In section III we introduce the linearly interpolated hierarchical *n*-gram language models (LIHLMs). We study the properties and parameters defining this model to show how it leads to better

estimate the probability of *n*-gram events. Section IV describes the backoff hierarchical class *n*-gram language modeling approach (HCLMs). The goal is to compute its performance to the performance of LIHLMs reported in section III. Section V presents the technique we use in building the class hierarchy. Section VI reports the data we used for training and evaluation and section VII describes the conducted experiments where we confirm the effectiveness of our approach to estimate the likelihood of n-gram events. Section VIII concludes the chapter.

2. Previous works

The idea of using classes to estimate the probability of unseen events in a backoff word *n*-gram model was proposed by several scientists [14], [15]. The basic principle of these approaches is to estimate the likelihood of unseen *n*-grams based on the class *n*-gram model and then, if needed, the (n-l)-grams. The originality of our approach is to use a hierarchical representation of the classes rather than an unstructured set of classes.

L. Bahl *et al.* proposed a tree-based statistical language model [16] where a linear interpolation is used to smooth the relative frequency at each node of the tree. L. Bahl *et al.* use information theoretic measures to construct equivalence classes of the context to cope with data sparseness. Using the approach of L. Bahl *et al.,* the likelihood of an *n*-grams *(wi,... w_n)* is computed as the linear interpolation of several word class language models extracted from the word class tree. P. Heeman investigated a similar hierarchical language modeling approach where POS tags, word identities, and a decision tree technique are used to estimate the probability distribution allowing generalization between POS tags and words [17]. Their tree-building strategy is based on the approach of L. Bahl *et al.* [16], and uses Breiman's decision tree learning algorithm [18] to partition the context into equivalence classes. P. Heeman uses a *binary* tree where at each node the clustering algorithm find a question about the POS tags and left context word identities in order to partition the node into 2 leaves. The approaches proposed by L. Bahl *et al.* in [16] and P. Heeman in [17] have some similarity with the technique of LIHLMs we propose. The main difference between LIHLMs and the approaches proposed by L. Bahl *et al.* in [16] and P. Heeman in [17] is in the technique we use for the interpolation scheme, the way we select active nodes, and the approach we use to build the class hierarchy. Also, the LIHLMs don't use POS information.

In 2003, one year after we introduced the hierarchical class n-gram approach [19], J. Bilmes and K. Kirchhoff published a hierarchical language model [20] where the likelihood of a word given its context is computed based on a vector of factors, instead of the word history. Factors may represent morphological classes, stems, etc. In the approach we propose, we do not use syntactic and morphological features. One advantage of our approach compared to those cited before is the use of a data-driven method to build the class hierarchy, which eliminate the costly *decision* tree build step.

P. Dupont and R. Rosenfeld proposed a multi-dimensional lattice approach where the likelihood of a word given a history is estimated based on multi-dimensional hierarchies. The strength of their approach lies in its generality and in the dynamic selection of a small subset of predictor contexts. An interpolation between these predictor contexts is then used to estimate the likelihood of a word given a history. However, the selection of these predictor nodes is still an open problem, which makes their approach difficult to use in real ASR applications. As stated by P. Dupont and R. Rosenfeld in [21], the reported results are preliminary and are based on perplexity only. The HCLMs we proposed in [9] shares some

similarities with the two-dimensional lattice technique of P. Dupont and R. Rosenfeld [21], where the first dimension is the length of the history equivalence class and the second dimension is the position in a word class hierarchy. Compared to P. Dupont and R. Rosenfeld, HCLMs do not need to select a subset of predictor contexts. Instead, HCLMs use the backoff technique and the most specific class to balances generalization ability and word specificity when estimating the likelihood of a word given a history. This makes HCLMs less complex and easy to integrate in real-time ASR applications.
Recently, another tree-based language modeling approach is proposed by P. Xu and F. Jelinek [22]. It explores the use of Random Forests in the structured language model, which uses rich syntactic information in predicting the next word based on words already seen. The goal is to construct Random Forests by randomly growing decision trees using syntactic information. Random Forests are a combination of decision tree classifiers originally developed for classification purposes.

3. Linearly interpolated hierarchical n-gram language models

The conditional probability of a word w given a history h, $P(w|h)$, is in general obtained by combining two components: a discounting model and a redistribution model. Discounting is related to the zero-frequency estimation problem [23]. The idea behind discounting is that a probability for all the words never observed after the history h must be estimated by discounting the n-gram relative frequency:

$$fr(w|h) = \frac{N(hw)}{N(h)} \tag{1}$$

where $N(.)$ denotes the frequency of the argument in the training data. By definition $N(h) = 0$ implies $fr(w|h) = 0$. Discounting produces a discounted conditional frequency $fr^*(w|h)$, such that:

$$0 \leq fr^*(w|h) \leq fr(w|h) \tag{2}$$

The zero-frequency probability $\lambda(h)$ is then defined as follows:

$$\lambda(h) = 1 - fr^*(w|h) \tag{3}$$

The zero-frequency probability $\lambda(h)$ is redistributed among the set of words never observed in the context h. Redistribution of the probability $\lambda(h)$ is performed proportionally to a more general distribution $P(w|h')$, where h' denotes a more general context.
Using the linear interpolation smoothing technique [24], [2], the conditional probability of a word w given a history h, $p(w|h)$, is estimated as follows:

$$P(w|h) = (1 - \lambda(h))\ fr(w|h) + \lambda(h)P(w|h') \tag{4}$$

where the same scheme applies to the lower-order distribution $P(w|h')$. The $\lambda(h)$ are such that $0 < \lambda(h) \leq 1$ if $N(h) > 0$, and $\lambda(h) = 1$ otherwize. The interpolation parameter is estimated using the expectation maximization algorithm [25].

When using classical linearly interpolated word *n*-gram models, typically the more general (n-l)-gram distribution is used to estimate the n-gram distribution. We recursively estimate the (n-k)-gram distribution using (n-k-l)-gram distribution until we reach the uniform distribution. Linear interpolation can be seen as a general smoothing approach that allows the combination of an arbitrary number of distribution or even language models. The most known and original version of the linear interpolated trigram (3-gram) language model [1] was not defined recursively as described in equation 4. It was presented as a linear combination of all order empirical distributions:

$$P(w_i|h) = P(w_i|w_{i-2}, w_{i-1}) =$$
$$\lambda_1(w_{i-2}, w_{i-1})\ fr(w_i|w_{i-2}, w_{i-1})\ +\ \lambda_2(w_{i-2}, w_{i-1})\ fr(w_i|w_{i-1}) + \quad (5)$$
$$\lambda_3(w_{i-2}, w_{i-1})\ fr(w_i)\ +\ \lambda_4(w_{i-2}, w_{i-1})$$

where $\lambda_i(h) \geq 0$ (*i* = 1, 2, 3, 4) and $\sum_i \lambda_i(h) = 1$.
Hence, using a recursive representation, the classical linearly interpolated word *n*-gram language models estimate the conditional probability of a word w given a history h, $P(w|h)$, according to the (n-l)-gram distribution:

$$P(w_i|h) = P(w_i|w_{i-n+1}^{i-1}) =$$
$$(1 - \lambda(w_{i-n+1}^{i-1}))fr(w_i|w_{i-n+1}^{i-1}) + \lambda(w_{i-n+1}^{i-1})P(w_i|w_{i-n+2}^{i-1}) \quad (6)$$

To better explore the power of word n-grams for frequent events and the predictive power of class *n*-grams for unseen or rare events, we propose the linearly interpolated hierarchical *n*-gram language models (LIHLMs). LIHLMs combine discounting and redistribution according to the linear interpolation smoothing technique [24], [2]. These models estimate the conditional probability of an *n*-gram (w_{i-n+1}^{i}), $P(w_i|w_{i-n+1}^{i-1})$ according to more general distribution extracted from the class hierarchy: we use the class of the most distant word w_{i-n+1} followed by the other words:

$$F(w_{i-n+1}), w_{i-n+2}^{i-1}$$

The function *F(x)* represents the class (parent) of *x* within the hierarchical word tree, where *x* can be a class itself, or a single word, depending on its location in the tree (cf. Section V). Let F_i^j denote the *j*th parent of word w_i:

$$F_i^j = F^{(j)}(w_i)$$

The probability $P(w_i|w_{i-n+1}^{i-1})$ is estimated as follows:

$$P(w_i|w_{i-n+1}^{i-1}) =$$
$$(1 - \lambda(w_{i-n+1}^{i-1}))fr(w_i|w_{i-n+1}^{i-1}) + \lambda(w_{i-n+1}^{i-1})P(w_i|F_{i-n+1}^{1}, w_{i-n+2}^{i-1}) \quad (7)$$

where $P(w_i|F^j_{i-n+1}, w^{i-1}_{i-n+2})$ is recursively estimated according to more general distribution by going up one level at a time in the hierarchical word clustering tree:

$$P(w_i|F^j_{i-n+1}, w^{i-1}_{i-n+2}) = \begin{cases} (1-\lambda(F^j_{i-n+1}, w^{i-1}_{i-n+2}))fr(w_i|F^j_{i-n+1}, w^{i-1}_{i-n+2}) + \\ \lambda(F^j_{i-n+1}, w^{i-1}_{i-n+2})P(w_i|w^{i-1}_{i-n+2}) \\ \qquad \textbf{if } F^{j+1}_{i-n+1} \textbf{ is the root} \\ (1-\lambda(F^j_{i-n+1}, w^{i-1}_{i-n+2}))fr(w_i|F^j_{i-n+1}, w^{i-1}_{i-n+2}) + \\ \lambda(F^j_{i-n+1}, w^{i-1}_{i-n+2})P(w_i|F^{j+1}_{i-n+1}, w^{i-1}_{i-n+2}) \\ \qquad \textbf{otherwise} \end{cases} \quad (8)$$

As a result, the whole procedure provides a consistent way to compute the probability of any *n*-gram event by exploring the classes that are in the hierarchical word tree. If the parent of the class F^j_{i-n+1} (respectively, the word w_{i-n+1}) is the class root, the context becomes the last (n-2) words, which is similar to the traditional linearly interpolated word *n*-gram models as described in equation 6.

Based on this definition, the linearly interpolated hierarchical n-gram approach is a generalization of the classical linearly interpolated word n-gram language models: word *n*-gram language models can be seen as linearly interpolated hierarchical *n*-gram language models with a single level (leaves) in the hierarchical word tree.

4. Backoff hierarchical class n-gram language models

The backoff hierarchical class *n*-gram models are introduced in [9]. The goal of this section is to briefly describe this approach in order to compare its performance to the performance of the linearly interpolated hierarchical *n*-gram language models.

When using the backoff hierarchical class *n*-gram models, the conditional probability of an unseen *n*-gram $P(w_i|w^{i-1}_{i-n+1})$ is estimated according to a more specific context than the (n-l)-gram $P(w_i|w^{i-1}_{i-n+2})$ We use as context the class of the most distant word w_{i-n+1} followed by the other words:

$$F(w_{i-n+1}), w^{i-1}_{i-n+2}$$

We remind that *F(x)* denotes the class (parent) of *x* within the hierarchical word tree.

The probability $P(w_i|w^{i-1}_{i-n+1})$ is estimated as follows:

$$P(w_i|w_{i-n+1}^{i-1}) = \begin{cases} \tilde{P}(w_i|w_{i-n+1}^{i-1}) & \textbf{if } N(w_{i-n+1}^{i}) > 0 \\ \alpha'(w_{i-n+1}^{i-1})P(w_i|F_{i-n+1}^{1}, w_{i-n+2}^{i-1}) & \textbf{otherwise} \end{cases} \quad (9)$$

where F_i^j as stated before denotes the j^{th} parent of word w_i, $N(.)$ denotes the frequency of the argument in the training data and $\alpha'(w_{i-n+1}^{i-1})$ is a normalizing constant guaranteeing that all probabilities sum to 1 [13]:

$$\alpha'(w_{i-n+1}^{i-1}) = \frac{1 - \sum_{w_i:N(w_{i-n+1}^{i})>0} P(w_i|w_{i-n+1}^{i-1})}{1 - \sum_{w_i:N(w_{i-n+1}^{i})>0} P(w_i|F(w_{i-n+1}), w_{i-n+2}^{i-1})} \quad (10)$$

The $\tilde{P}(.)$ in equation 9 is estimated as follows:

$$\tilde{P}(w_i|w_{i-n+1}^{i-1}) = d_{N(w_{i-n+1}^{i})} \frac{N(w_{i-n+1}^{i})}{N(w_{i-n+1}^{i-1})} \quad (11)$$

where the term $d_{N(.)}$ denotes the Turing's discounting coefficient [13].

If the event $F_{i-n+1}^{j}, w_{i-n+2}^{i}$ is not found in the training data ($N(F_{i-n+1}^{j}, w_{i-n+2}^{i}) = 0$), we recursively use a more general context by going up one level at a time in the hierarchical word clustering tree. This context is obtained by taking the parent of the first class in the hierarchy, followed by the $n-2$ last words:

$$P(w_i|F_{i-n+1}^{j}, w_{i-n+2}^{i-1}) = \begin{cases} \tilde{P}(w_i|F_{i-n+1}^{j}, w_{i-n+2}^{i-1}) & \textbf{if } N(F_{i-n+1}^{j}, w_{i-n+2}^{i}) > 0 \\ \alpha'(F_{i-n+1}^{j}, w_{i-n+2}^{i-1})P(w_i|w_{i-n+2}^{i-1}) & \textbf{if } F_{i-n+1}^{j+1} \textbf{ is the root} \\ \alpha'(F_{i-n+1}^{j}, w_{i-n+2}^{i-1})P(w_i|F_{i-n+1}^{j+1}, w_{i-n+2}^{i-1}) & \textbf{otherwise} \end{cases} \quad (12)$$

where the normalizing constant $\alpha'(F_{i-n+1}^{j}, w_{i-n+2}^{i-1})$ is computed as follows to guarantee that all probabilities sum to 1:

$$\alpha'(F_{i-n+1}^{j}, w_{i-n+2}^{i-1}) = \begin{cases} \dfrac{1-\sum_{w_i:N(F_{i-n+1}^{j}, w_{i-n+2}^{i})>0} P(w_i|F_{i-n+1}^{j}, w_{i-n+2}^{i-1})}{1-\sum_{w_i:N(F_{i-n+1}^{j}, w_{i-n+2}^{i})>0} P(w_i|w_{i-n+2}^{i-1})} & \textbf{if } F_{i-n+1}^{j+1} \textbf{ is the root} \\ \dfrac{1-\sum_{w_i:N(F_{i-n+1}^{j}, w_{i-n+2}^{i})>0} P(w_i|F_{i-n+1}^{j}, w_{i-n+2}^{i-1})}{1-\sum_{w_i:N(F_{i-n+1}^{j}, w_{i-n+2}^{i})>0} P(w_i|F_{i-n+1}^{j+1}, w_{i-n+2}^{i-1})} & \textbf{otherwise} \end{cases} \quad (13)$$

The procedure provides a consistent way to compute the probability of rare or unseen *n*-grams by backing-off along the classes that are defined in the hierarchical word tree. If the parent of the class F_{i-n+1}^{j} (respectively, the word w_{i-n+1}) is the class root, the context becomes the last $n-2$ words, which is similar to the traditional back-off word *n*-gram model [13]. Word *n*-gram language models are the backoff hierarchical class *n*-gram language models with a single level in the hierarchical word tree.

5. Hierarchical word clustering algorithm

The hierarchical approaches we propose relies on the design of a classifier that allows finding a parent (class) for a word *w*. It is important to note that the two hierarchical *n*-gram languages describes in the two previous sections are able to integrate any classification approach. A better classifier will lead to a more accurate hierarchical model. The Maximum Mutual Information (MMI) clustering algorithm proposed by P. Brown *et al.* in [26] and by F. Jelinek in [1] has been widely adopted. Using the MMI approach, the computation required to obtain a language model with *C* classes using a vocabulary of *V* words is in the order of V^3. A greedy merge method is also used, based on the MMI theory, and requires an order of V^2C operations. Several iterations of the algorithm can be performed to improve performance. However, when using large vocabularies, this becomes quickly intractable given the computational complexity [27]. On the other hand, the minimum discriminative information (MDI) clustering approach proposed by S. Bai *et al.* gives similar results as the MMI method, while dramatically reducing the computation [27], as it only involves computing less than V^2 logarithms; for comparison results between MDI and MMI approaches, readers may refer to [27]. Consequently, in our approach we adopted the clustering technique of S. Bai *et al.*, which is based on minimum discriminative information.

The hierarchical word clustering algorithm proceeds in a top-down manner to cluster a vocabulary word set *V*, and is controlled by two parameters: (1) the maximum number of descendant nodes (clusters) *C* allowed at each node, (2) the minimum number of words *K* in one class O_c: ($N(O_c) \geq K$). In our case, *K* is set to 2. Starting at the root node, which contains a single cluster representing the whole vocabulary, we compute the centroid o_i, of the entire space (word set). An initial codebook is then built by assigning the *C* closest words to o_i, into *C* clusters, which define the immediate child nodes of the root node [9]. The

process is continued recursively on each descendant node to grow the tree. The algorithm stops when a predefined number of levels (depth) is reached or when the number of proposed clusters for one node O_c is equal to 1 [10]. Each word in the vocabulary constitutes a leaf in the tree, words are clustered into classes, and classes are recursively clustered into more general sets of classes, until the root. At the top of the tree, the root node is a class containing all the words in the vocabulary. A summary of the minimum discriminative information methods is presented in the next section, followed by a detailed description of the word clustering algorithm.

A. Minimum Discriminative Information

The left and right contexts are used to cluster a word set, meaning that words occurring frequently in similar contexts should be assigned to the same class. The contextual information of a word w, $p_d\{w\}$, is estimated by the probability of w given its d left and right neighboring words. Given a set of V words, in the case of $d = 1$, we have:

$$p_1\{w\} = \{p_1 l\{w\}, p_1 r\{w\}\} \tag{14}$$

The terms $p_1 l\{w\}$ and $p_1 r\{w\}$ denote respectively the left-bigram and right-bigram contextual information of word w, given a vocabulary of V words $\{w_v\}_{v=1}^{V}$:

$$p_1 l\{w\} = \{pl(w_1|w), \ldots pl(w_V|w)\} \tag{15}$$

and

$$p_1 r\{w\} = \{pr(w_1|w), \ldots pr(w_V|w)\} \tag{16}$$

The clustering algorithm is based on two principles: (1) words with similar contexts are merged into the same cluster; (2) a word cluster is built according to the cluster of its neighboring words (contextual information). The contextual information of word w, $p\{w\}$, is represented by the probability of w given its right and left context bigrams. The problem is then to define the similarity of two words in terms of their contextual information. To define the similarity of two words w_1 and w_2 in terms of their contextual information, we use the Kullback-Leibler distortion measure $D(w_1, w_2)$ [6]:

$$\begin{aligned} D(w_1, w_2) &= \sum_{v=1}^{V} pl(w_v|w_1) \ \ log \ \frac{pl(w_v|w_1)}{pl(w_v|w_2)} \\ &+ \sum_{v=1}^{V} pr(w_v|w_1) \ \ log \ \frac{pr(w_v|w_1)}{pr(w_v|w_2)} \end{aligned} \tag{17}$$

which is also known as relative entropy. The objective of partitioning the vocabulary is to find a set of centroids $\{o_c\}$ for clusters $\{O_c\}$, $c = 1, \ldots C$ that leads to the minimum global discriminative information:

$$
\begin{aligned}
GDI &= \sum_{c=1}^{C} \sum_{i \in O_c} D(w_i, o_c) \\
&= \sum_{i=1}^{V} \sum_{v=1}^{V} pl(w_v|w_i) \ \ log \ pl(w_v|w_i) \\
&+ \sum_{i=1}^{V} \sum_{v=1}^{V} pr(w_v|w_i) \ \ log \ pr(w_v|w_i) \\
&- \sum_{c=1}^{C} \sum_{i \in O_c} \sum_{v=1}^{V} pl(w_v|w_i) \ \ log \ pl(w_v|o_c) \\
&- \sum_{c=1}^{C} \sum_{i \in O_c} \sum_{v=1}^{V} pr(w_v|w_i) \ \ log \ pr(w_v|o_c) \\
&= B(w) - R(w).
\end{aligned} \tag{18}
$$

The term *B(w)* is a constant independent of the partitioning. Hence, *R(w)* is maximized when the global discriminative information is minimized. Each cluster O_c is represented by a centroid o_c. According to equation 14, $p_1\{w\}$ is defined as a vector of dimension $2 \bullet V$, whose first V components are based on the left-context bigrams, and last V components are based on the last V right-context bigrams. For simplicity, let us drop the left/right indices, and represent $p_1\{w\}$ as follows:

$$p_1\{w\} = \{p(k|w), \ k = 1, \ldots 2V\} \tag{19}$$

Given equation 19, the centroid of the class $O_c = \{wi, i = 1 \ldots v_c\}$ is estimated as follows [28], [29]:

$$o_c = \{o(k|o_c), \ k = 1, \ldots 2V\}$$

where $o(k|o_c)$ is approximated by [27]:

$$o(k|o_c) = \frac{1}{v_c} \sum_{i=1}^{v_c} p(k|w_i) \tag{20}$$

B. Word Clustering Algorithm

We present in this section how to classify a word set into *C* classes, under the constraint that at least *K* words should appear in each class O_c. Our approach is based on the **K**-means clustering technique [30], [31], where we define centroids and distances specific to words.

We start at the root note by computing the centroid o_i, of the entire space (word set). An initial codebook is then built by assigning the *C* closest words to o_i, into *C* clusters. The centroids of each cluster are then re-computed, and the process is iterated until the average distortion GDI converges. This process is then recursively applied. The pseudo-code of the algorithm is as follows:

- step 1: start with an initial codebook;
- step 2: for each w_i, $i = 1,\ldots V$,

    - find the closest class O to w_i using Kullback-Leibler distortion measure and add w_i to it [19].
- step 3: update the codebook using the minimum distance or nearest neighbor rule [9], [28];
- step 4: **if** *GDI* > t **then** go to step 2
    - where t is an experimentally tuned threshold controlling the convergence of the process; the current set of clusters may leads to the minimum global discriminative information (cf. equation 18).
- step 5: **if** $\exists\ O_c\ /\ N(O_c) < K$ **then** $(C \leftarrow C - 1)$ and go to 1**, otherwise** stop.

Step 5 is necessary since it is used to control the number of words in a class: if there are too few words in a class *($N(O_C)$ < K),* that class is merged with another one. In practice, only a few iterations of the algorithm are required to achieve fairly good results [27]. Since each word is characterized by the contextual statistical vector $p_d\{w\}$, the centroid of each class is easily found using equation 20. The advantage of this algorithm is its simplicity in finding centroids; the cost of merging words or classes becomes less expensive. Once *C* classes have been defined, the algorithm is recursively applied within each class to grow the tree.

6. Corpus

Experiments are performed on the Wall Street Journal 94-96 text corpus. This database is divided into training, development and test sets. For language modeling purposes, the training set contains 56 million words, and the test set contains approximately 6 million words. A development set of 5 million words is also used to tune the different parameters of the model, including the depth of the clustering tree. Two vocabulary sizes are used: a first one containing 5,000 words *(5K)* and a second one including 20,000 words *(20K).* Note that the *5K* vocabulary leads to about 2% of out-of-vocabulary words on the test data, and in that regard differs substantially from the official WSJ *5K* lexicon that was designed for a closed-set evaluation (no OOV words). The *20K* vocabulary has a 1.1% out-of-vocabulary rate on the test data. In our experiments, we use *open* vocabulary where the unknown word is part of the model [1].

7. Experiments

Our objective is to show that the use of word class hierarchy in language modeling better handle the likelihood estimation of n-gram events. We show in this section the performance of linearly interpolated hierarchical n-gram language models as well as the performance of the backoff hierarchical class n-gram language models. We compare the performance of these two techniques to the performance of commonly used methods such as the linearly interpolated n-gram language models (LILMs) and the backoff n-gram language models.

Performance is evaluated in terms of test perplexity and word error rate (WER) using Bell Labs' speech recognizer [32]. Both HCLMs and the backoff word n-gram LMs use the backing-off smoothing technique to estimate the likelihood of unseen events [13]. Also, both LIHLMs and LILMs combine discounting and redistribution according to the linear interpolation smoothing technique [24], [2].

As a reminder, HCLMs and LIHLMs with a number of levels in the class hierarchy equal to 0 are in fact the classical backoff word n-gram LMs and LILMs respectively. Hence, we believe that it is fair to consider both backoff word n-gram LMs and LILMs as baselines for

comparison purpose. We also report in this section comparison results with word class n-gram (n-class) LMs as well as a linear interpolation between word n-gram and n-class LMs [1]. In addition, we investigate how the number of levels defined in the class hierarchy impacts the performance of our approach.

A. Perplexity Experiments

Perplexity is typically used to measure the performance of language models. It is therefore interesting to look at the perplexity obtained by the two hierarchical n-gram models for different number of levels in the hierarchy. The number of levels in the hierarchy L represents the depth of the word class tree. The maximum number of direct descendant of a class is fixed to $C = 6$ (cf. section 5). Experiments carried out with different values of C led to similar results [9]. The *maximum* number of classes generated when building the class tree is $\sum_{l=1}^{L} C^l$: e.g., for a word class tree of two levels, the root node is split into a maximum of C classes and each class is split into C other classes, leading to a maximum of C^2 clusters at the second level. This number is optimized at each level of the hierarchy by the classification algorithm in order to converge to an optimum (cf. section 5-B).

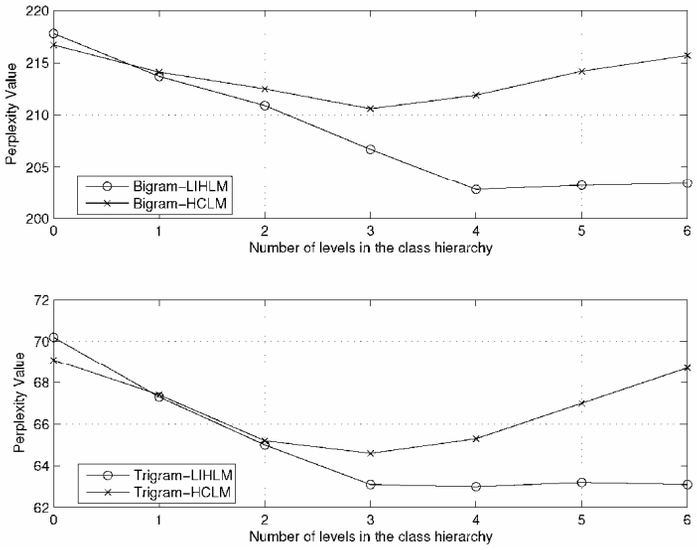


Figure 1. Trigram and bigram test perplexity on WSJ using 5K vocabulary with different number of levels in the class hierarchy

Figure 1 presents the performance of LIHLMs, HCLMs, LILMs and backoff word n-gram LMs when the 5K vocabulary is used. We remind that LILMs and backoff word n-gram LMs are respectively the LIHLMs and HCLMs with a number of levels in the class hierarchy equal to 0. Experimental results show that we do not need a large number of levels in the class hierarchy to improve upon the baseline: three or four levels are enough to achieve a good performance compare to baseline models. When using the *5K* vocabulary, trigram LIHLM improves the baseline backoff word trigram language model by 10% (63.0 vs. 69.1). However, a very slight improvement of 3% in terms of perplexity is obtained by the trigram LIHLM when compared to trigram HCLM (63.0 vs. 64.6). A similar behavior is obtained for bigram events: a 6% improvement of the test perplexity on the whole test set is observed

(97.2 for bigram LIHLM vs. 103.0 for backoff word bigram model). A very small improvement of 3% is also obtained by the bigram LIHLM when compared to bigram HCLM (97.2 vs. 100.3).

Performance in terms of perplexity when using the *20K* vocabulary is presented in Figure 2. Results in Figure 2 again show the effectiveness of the LIHLMs in improving the perplexity value of the backoff word n-gram LMs: 7% improvement for bigrams (202.8 vs. 216.7) and 10% improvement for trigrams (127.5 vs. 140.8). The LIHLMs also effectively improves the performance of HCLMs by 4% for both bigrams (202.8 vs. 210.6) and trigrams (127.5 vs. 132.2).

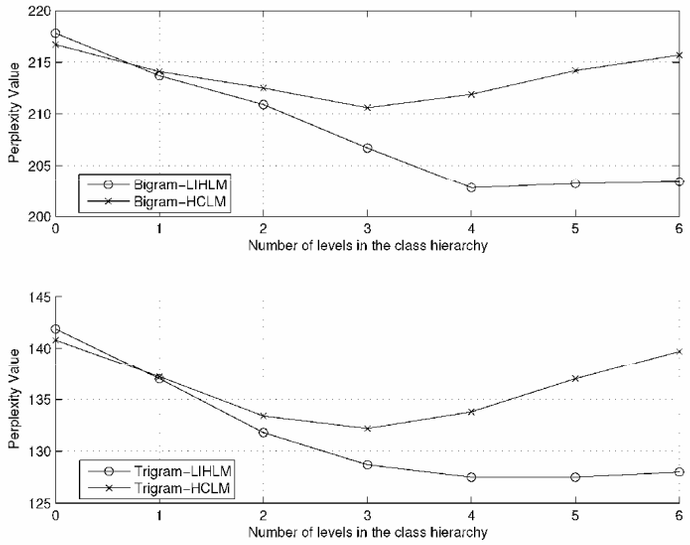


Figure 2. Trigram and bigram test perplexity on WSJ using 20K vocabulary with different number of levels in the class hierarchy

For both the *5K* and *20K* vocabularies, we notice that backoff word n-gram LMs are doing slightly better that classic linearly interpolated LMs. We believe that the difference is statistically insignificant, which will not allow us to draw any conclusion. Another important point to mention is that in both the *5K* and *20K* vocabularies, the perplexity value of the HCLMs decreased for the first three levels in the class hierarchy and then it starts to increase. This observation is not true for the LIHLMs, where the perplexity value achieves its optimum with four levels in the class hierarchy and doesn't increase afterward. One may conclude that, compared to HCLMs, LIHLMs are less sensitive to the depth of the tree (number of levels in the class hierarchy).

Results reported in Figure 1 and Figure 2 are computed on the test data. We observed the same behavior on the development set: with (C = 6, the best performance in terms of perplexity on the entire development set is obtained with a maximum level in the class hierarchy set to L = 3. On 20K vocabulary, trigram perplexity of HCLMs decreased from 142.4 for L = 0 (i.e., baseline backoff word n-gram LMs) to 134.6 for L = 3 and then it starts to increase. The same behavior is observed on the 5K vocabulary: trigram perplexity of HCLMs decreased from 71.0 for L = 0 to 66.7 for L = 3 and then starts to increase.

B. Comparison with Word Class n-gram Language Models

As stated in the introduction, one of the approaches that can overcome the probability estimation problem of unseen n-grams event is the class *n*-gram language models [1], [8]. For sparse data, class n-gram language models usually generalize better on unseen *n*-grams than standard word-based language models. Nevertheless, for large training corpus, word *n*-gram LMs are still better in capturing collocational relations between words. To confirm this point, we built word class *n*-gram (*n*-class) LMs and compared their performance to the baseline *n*-gram models as well as the hierarchical approaches. We also investigate a comparison results with linear interpolation of word *n*-gram and *n*-class LMs. In the backoff *n*-class models, the conditional probability of the *n*-gram $w_{i-n+1}^{i} = w_{i-n+1}, \ldots w_i$, is estimated as follows:

$$P(w_i|w_{i-n+1}^{i-1}) = P(w_i|F(w_{i-n+1}), \ldots F(w_{i-1})) \tag{21}$$

where the function *F(x)* represents the class of *x*. As stated by J. Goodman in [33], equation 21 showed to give stronger class model than estimating the conditional probability as follows:

$$P(w_i|w_{i-n+1}^{i-1}) = P(w_i|F(w_i)) \quad P(F(w_i)|F(w_{i-n+1}), \ldots F(w_{i-1})) \tag{22}$$

Notice that the hierarchical class n-gram LMs is able to integrate any classification approach for building the class hierarchy. In order to make a fair comparison between the proposed hierarchical approach and the backoff n-class LMs, we *should* use the same classification technique. Hence, to build the class set, we use the MDI approach (cf. section 5-B) that assigns each word to a unique class. We initialize the MDI classifier with a maximum number of classes equal to 1200 assuming that one class should contains at least 5 words. We present in table I, the perplexity values obtained by the different LMs on the entire test set. The hierarchical approach uses a maximum number of direct descendant of a class fixed to C = 6 and a number of levels in the hierarchy set to L = 3; both values are tuned on the development set (cf. section 7- A).

	5K-WSJ	**20K-WSJ**
Bigram LMs		
Class	115.2	228.9
Word (Baseline)	103.0	216.7
LI (Word + Class)	102.2	215.8
HCLM	100.3	210.6
LIHLM	97.2	202.8
Trigram LMs		
Class	76.5	154.0
Word (Baseline)	69.1	140.8
LI (Word + Class)	68.2	138.6
HCLM	64.6	132.2
LIHLM	63.0	127.5

Table 1. Perplexity on WSJ of world class *n*-gram LMs (class), word *n*-gram LMs (word), linear interpolation of word *n*-gram and *n*-class LMS (LI), hierarchical class *n*-gram LMS (HCLM), and linearly interpolated hierarchical *n*-gram LMs (LIHLM)

As expected, the perplexity of the baseline word n-gram LMs is better than the class word n-gram LMs: 216.7 vs. 228.9 for the bigram model and 140.8 vs. 154.0 for the trigram model with the 20K vocabulary (similar behavior was observed with the 5K vocabulary). Also, compared to the baseline word n-gram LMs, we notice that a linear interpolation of word n-gram LMs and n-class LMs doesn't led to a considerable improvement (215.8 vs. 216.7 for bigram and 138.6 vs. 140.8 for trigram on 20K vocabulary). On both bigram and trigram, results show that the proposed hierarchical LMs outperform the other approaches.

C. Speech Recognition Experiments

For ASR experiments, the word error rate (WER) on the 5K WSJ has been evaluated on the 330 sentences of the si_et_05 evaluation set. The 333 sentences of the si_et_20 evaluation set were used for the 20K ASR experiment. We used tied-state triphone acoustic models built on the WSJ SI-84 database. The speech recognition experiments were performed using the Bell Labs ASR system [32]. The ASR system is based on a Viterbi algorithm, where at each node in the search the acoustic score is interpolated with the language modeling score. Hence, we do not need to modify the decoder structure in order to integrate LIHLMs and HCLMs: we only replaced the language modeling score, initially estimated using the trigram language model, with the score estimated using LIHLMs and HCLMs respectively. Once the language model is integrated to the ASR system, the pruning parameters are re-computed to boost its accuracy. We gave equivalent setting to the pruning parameters to make sure that the decoder search doesn't favor one model over another.

Recall that the *5K* vocabulary differs from the official WSJ *5K* lexicon which was designed for a closed-set evaluation. Results presented in Table II show that there is no significant improvement in performance between the baseline backoff bigram model, bigram HCLM, and bigram LIHLM. These results can be explained by the small number of unseen bigrams in this experimental setup and therefore the lack of room for any significant improvement: unseen bigrams constitute 4% and 8% of the total bigrams for the *5K* and *20K* vocabularies respectively. However, when the trigram model is used, the number of unseen events increases to 27% for the *5K* vocabulary and to 34% for the 20K vocabulary, leading to 12% and 10% reduction of the WER, respectively. We also note that HCLMs and LIHLMs have the same ASR performance. Compared to the recognizer using linear interpolation between word and class trigram model, the use of hierarchical approaches improves performance by 6% (11.1% vs. 12.0%) and by 8% (6.7% vs. 7.3%) relative for WSJ-20K and WSJ-5K respectively.

	5K		20K	
	bigram	trigram	bigram	trigram
Baseline	9.3%	7.6%	14.2%	12.4%
LI (word + class)	9.2%	7.3%	14.1%	12.0%
HCLM	9.0%	6.7%	13.9%	11.2%
LIHLM	9.0%	6.7%	13.8%	11.1%

Table 2. **WER** on *5K* and *20K* vocabularies using word *n*-gram (baseline), linear interpolation between word and class *n*-gram (LI), hierarchical class *n*-gram (HCLM), and linearly interpolated hierarchical *n*-gram LMS (LIHLM) respectively

We think that the effectiveness of our approach may also depend on the quality of the acoustic model and the domain on which the recognizer is employed. For instance, if the

language model has very low perplexity on unseen events and if the acoustic model is able to well discriminate words under these unseen context, then a big portion of the errors made by the recognizer are more likely to be accumulated on frequent context. The opposite is also true: if the language model has low perplexity on frequent events and the acoustic model is able to discriminate words under these frequent events, then errors are more likely to be accumulated on low acoustic certainty. Hence, similarly to [9], [10], we would like to raise the fact that we may not need to improve the perplexity on the whole data in order to reduce the word error rate of ASR systems. It may be sufficient to reduce the perplexity of unseen events rather than the frequently seen events, since ASR systems are more sensitive to unseen events.

8. Conclusion

We have investigated a new language modeling approach called linearly interpolated *n*-gram language models. We showed in this chapter the effectiveness of this approach to estimate the likelihood of *n*-gram events: the linearly interpolated *n*-gram language models outperform the performance of both linearly interpolated *n*-gram language models and backoff *n*-gram language models in terms of perplexity and also in terms word error rate when intergrated into a speech recognizer engine. Compared to traditional backoff and linearly interpolated LMs, the originality of this approach is in the use of a class hierarchy that leads to a better estimation of the likelihood of *n*-gram events. Experiments on the WSJ database show that the linearly interpolated *n*-gram language models improve the test perplexity over the standard language modeling approaches: 7% improvement when estimating the likelihood of bigram events, and 10% improvement when estimating the likelihood of trigram events.

Speech recognition results show to be sensitive to the number of unseen events: up to 12% reduction of the WER is obtained when using the linearly interpolated hierarchical approach, due to the large number of unseen events in the ASR test set. The magnitude of the WER reduction is larger than what we would have expected given the observed reduction of the language model perplexity; this leads us to an interesting assumption that the reduction of unseen event perplexity is more effective for improving ASR accuracy than the perplexity associated with seen events. The probability model for frequently seen events may already be appropriate for the ASR system so that improving the likelihood of such events does not correct any additional ASR errors (although the total perplexity may decrease.) Thus, it may be that similar reductions of the perplexity are not equivalent in terms of WER improvement. The improvement in word accuracy also depends on the errors the recognizer makes: if the acoustic model alone is able to discriminate words under unseen linguistic contexts, then improving the LM probability for those events may not improve the overall WER.

Compared to hierarchical class *n*-gram LMs, we observed that the new hierarchical approach is not sensitive to the depth of the hierarchy. As future work, we may explore this approach with a more accurate technique in building the class word hierarchy.

9. References

F. Jelinek, Self-organized language modeling for speech recognition, *Readings in Speech Recognition, A. Waibel and K-F. Lee editors,* pp. 450-506, Morgan Kaufmann, San Mateo, Calif., 1990. [1]

Renato DeMori, Ed., *Spoken Dialogues with Computers,* Academic Press, 1998. [2]

V. Gupta, M. Lenning, and P. Mermelstein, A language model for very large vocabulary speech recognition, *Computer Speech and Language,* pp. 331-344, 1992. [3]

J. Bellegarda, Exploiting latent semantic information in statistical language modeling, *Proceedings of the IEEE,* vol. 88, no. 8, August 2000. [4]

R. Rosenfeld, Two decades of statistical language modeling: Where do we go from here?, *Proceedings of the IEEE,* vol. 88, no. 8, 2000. [5]

T.M. Cover and J.A. Thomas, *Elements of Information Theory,* Wiley and Sons, 1991. [6]

Z. Bi, C. Faloutsos, and F. Korn, The 'DGX distribution for mining massive, skewed data, in *Conference on Knowledge Discovery and Data Mining,* 2001. [7]

B. Suhm and W. Waibel, Towards better language models for spontaneous speech, in *Proc. ICSLP-1994,* 1994. [8]

I. Zitouni, Backoff hierarchical class n-gram language models: Effectiveness to model unseen events in speech recognition, *Journal of Computer Speech and Language, Academic Press,* vol. 21, no. 1, pp. 88-104, 2007. [9]

I. Zitouni, O. Siohan, and C-H. Lee, Hierarchical class n-gram language models: Towards better estimation of unseen events in speech recognition, in *Proc. Eurospeech-2003,* Geneva, Switzerland, 2003. [10]

I. Zitouni, Q. Zhou, and Q.P. Li, A hierarchical approach for better estimation of unseen event likelihood in speech recognition, in *Proc. IEEE NLPKE-2003,* Beijing, China, 2003. [11]

I. Zitouni and H.J. Kuo, Effectiveness of the backoff hierarchical class n-gram language models to model unseen events in speech recognition, in *Proc. IEEE ASRU-2003,* St. Thomas, US Virgin Islands, 2003. [12]

S.M. Katz, Estimation of probabilities from sparse data for the language model component of a speech recognizer, *IEEE Transactions on Acoustic, Speech, and Signal Processing,* vol. 35, no. 3, 1987. [13]

J.W. Miller and F. Alleva, Evaluation of a language model using a clustered model backoff, in *Proc. ICSLP-1996,* 1996. [14]

C. Samuelsson and W. Reichl, A class-based language model for large-vocabulary speech recognition extracted from part-of-speech statistics, in *Proc. ICASSP-1999,* 1999. [15]

L. Bahl, P. Brown, P. de Souza, and R. Mercer, A tree-based statistical language model for natural language speech recognition, in *IEEE Transaction on Acoustics, Speech and Signal Processing,* July 1987, vol. 37, pp. 1001-1008. [16]

P.A. Heeman, Pos tags and decision trees for language modeling, in *Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora,* College Park, Maryland, June 1999, pp. 129-137. [17]

L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees,* Wadsworth & Brooks, 1984. [18]

I. Zitouni, O. Siohan, H-K.J. Kuo, and C-H. Lee, Backoff hierarchical class n-gram language modelling for automatic speech recognition systems, in *Proc. ICSLP-2002,* Denver, USA, 2002. [19]

J.A. Bilmes and K. Kirchhoff, Factored language models and generalized parallel backoff, in *Proceeding of HLT/NAACL,* Canada, May 2003. [20]

P. Dupont and R. Rosenfeld, Lattice based language models, Tech. Rep. CMU-CS-97-173, Carnegie Mellon University, 1997. [21]

P. Xu and F. Jelinek, Random forests in language modeling, in *Conference on Empirical Methods in Natural Language Processing,* 2004. [22]

I.H. Witten and T.C. Bell, The zero-frequency problem: estimating the probabilities of novel events in adaptive text compression, *IEEE Transactions on Information Theory,* vol. 37, no. 4, pp. 1085-1094, 1991. [23]

F. Jelinek and R.L. Mercer, Interpolated estimation of markov source parameters from sparse data, in *Pattern Recognition in Practice,* Amsterdam, Holland, 1980, pp. 381-397. [24]

A.P. Dempster, N.M. Laird, and D.B. Rubin, Maximum likelihood from incomplete data via the em algorithm, *Journal of the Royal Statistical Society ser B,* vol. 39, pp. 1-38, 1977. [25]

P.P. Brown, V.J. DellaPietra, P.V. DeSouza, J.C. Lai, and R.L. Mercer, Class-based n-gram models of natural language, *Computational Linguistics,* vol. 18, no. 4, pp. 467-479, 1992. [26]

S. Bai, H. Li, Z. Lin, and B. Yuan, Building class-based language models with contextual statistics, in *Proc. ICASSP-1998,* 1998. [27]

H. Li, J.P Haton, J. Su, and Y. Gong, Speaker recognition with temporal transition models, in *Eurospeech-95,* Madrid, Spain, 1995. [28]

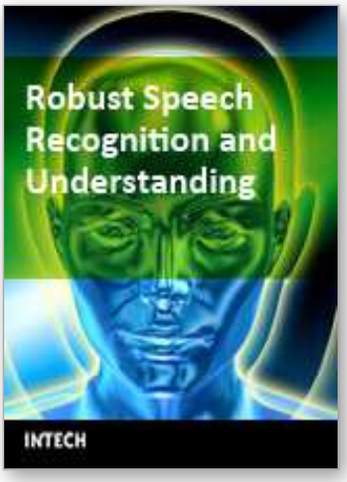
T.M. Cover and P.E. Hart, Nearest neighbor pattern classification, *IEEE Transaction on Information Theory,* vol. 13, no. 1, pp. 21-27, 1967. [29]

J. MacQueen, Some methods for classification and analysis of multi-variate observations, in *Proc. of the Fifth Berkeley Symp. on Math., Statistics and Probability, LeCam, L.M., and Neyman, J., (eds.), Berkeley: U. California Press, 281,* 1967. [30]

C. Darken and J. Moody, Fast adaptive k-means clustering: Some empirical results, in *Int. Joint Conf. on Neural Networks,* 1990, vol. 2, pp. 233-238. [31]

Q. Zhou and W. Chou, An approach to continuous speech recognition based on self-adjusting decoding graph, in *Proc. ICASSP-1997,* 1997, pp. 1779-1782. [32]

J. Goodman, A bit of progress in language modeling, *Computer Speech and Language,* pp. 403-434, October 2001. [33]



Robust Speech Recognition and Understanding

Edited by Michael Grimm and Kristian Kroschel

ISBN 978-3-902613-08-0

Hard cover, 460 pages

Publisher I-Tech Education and Publishing

Published online 01, June, 2007

Published in print edition June, 2007

This book on Robust Speech Recognition and Understanding brings together many different aspects of the current research on automatic speech recognition and language understanding. The first four chapters address the task of voice activity detection which is considered an important issue for all speech recognition systems. The next chapters give several extensions to state-of-the-art HMM methods. Furthermore, a number of chapters particularly address the task of robust ASR under noisy conditions. Two chapters on the automatic recognition of a speaker's emotional state highlight the importance of natural speech understanding and interpretation in voice-driven systems. The last chapters of the book address the application of conversational systems on robots, as well as the autonomous acquisition of vocalization skills.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen