Neural Network based Feature Extraction for Speech and Image Recognition

Von der Fakultät für
Mathematik, Informatik und Naturwissenschaften der
RWTH AACHEN UNIVERSITY

zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften
genehmigte Dissertation

vorgelegt von

Dipl.-Inform. Christian Plahl
aus Bielefeld

Berichter: Univ.-Prof. Dr.-Ing. Hermann Ney
Priv.-Doz. Dr.Ing. Björn W. Schuller

Tag der mündlichen Prüfung: 23. Januar 2014

Dipl.-Inform. Christian Plahl
Human Language Technology and Pattern Recognition Group
RWTH Aachen University
plahl@cs.rwth-aachen.de

Erklärung

Hiermit versichere ich, dass ich die vorliegende Doktorarbeit selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel verwendet habe. Alle Textauszüge und Grafiken, die sinngemäß oder wörtlich aus veröffentlichten Schriften entnommen wurden, sind durch Referenzen gekennzeichnet.

Aachen, Juni 2014

Dipl.-Inform. Christian Plahl

Abstract

This work investigates features derived from an artificial neural network. These artificial neural network based probabilistic features have become a major component of current state-of-the-art systems for automatic speech recognition and other areas, e.g. image recognition. A detailed study of the artificial neural network based features helps to improve the feature extraction and to understand which information of the speech signal is relevant for recognition.

Two algorithms are investigated which are widely used to integrate the information derived from an artificial neural network: the *tandem* and the *hybrid* approach. This work studies the effect of each of the algorithms in terms of recognition performance w.r.t. word error rate and the computational requirements. In addition, a detailed comparison and a discussion of the main advantages of each integration approach are given.

Furthermore, novel extensions are proposed improving the artificial neural network feature extraction and the final recognition performance of the systems trained. These extensions concern the input features and the topology of the network used to train the artificial neural network and are independent of the integration method. Different short-term and long-term features model other complementary aspects of the speech signal. By combining these different feature sets the development circle of the speech recognition system can be simplified. This allows increasing the model complexity of the artificial neural network or of the acoustic model.

The topology of an artificial neural network has a huge impact on the quality of the features derived from the artificial neural network. This work investigates the hierarchical framework, the bottle-neck processing and recurrent neural networks, especially the long-short-term-memory structure and the training of bi-directional networks. Furthermore, this work examines cross-lingual artificial neural network features and their impact on the topology and the amount of audio data used to train such features. The training and testing language of the artificial neural network features differs and the system development circle is simplified when such cross-lingual artificial neural network based features are used.

In addition, this work analyses different supervised and unsupervised weight pre-training techniques. The initialization of the weights of a deep neural network is critical since the optimization function is non-convex. A new unsupervised pre-training technique is developed

which allows the optimization of the loss function directly and provides a clear stopping criterion compared to other pre-training techniques like Restricted Boltzmann Machines.

Finally, this work analyzes the generality of the artificial neural network based feature extraction approach by transferring the concept to different image tasks, optical character recognition and automatic sign language recognition. While most results are confirmed, some surprising new results are obtained.

Zusammenfassung

Diese Doktorarbeit untersucht Merkmale, die mit Hilfe von künstlichen neuronalen Netzen erzeugt werden. Diese probabilistischen Merkmale sind als wichtiger Bestandteil aktueller automatischer Spracherkennungssysteme unverzichtbar geworden. Ebenso erfolgreich werden sie auch in vielen anderen Bereichen der Mustererkennung eingesetzt, wie z.B. in der Bildererkennung oder der Handschriftenerkennung. Durch eine detaillierte Analyse dieser Merkmale und ihres Entstehungsprozesses wird die Merkmalsgewinnung einerseits optimiert, andererseits lassen sich die für die automatische Spracherkennung relevanten Bestandteile des Sprachsignals identifizieren.

Der *hybride* und der *tandem* Ansatz werden in dieser Arbeit ausführlich untersucht. Beide Verfahren stellen dem Erkennungssystem die aus dem künstlichen neuronalen Netz gewonnene Information in unterschiedlicher Weise bereit. Die Verfahren werden hinsichtlich ihrer Erkennungsleistung und der benötigten Rechenleistung untersucht. Ihre wichtigsten Vor- und Nachteile werden gegenübergestellt und ihre Bedeutung für das Erkennungssystem diskutiert.

Um die Erkennung mit den aus dem neuronalen Netz gewonnenen Merkmalen zu verbessern, wird sowohl die Bedeutung der verwendeten Struktur des künstlichen neuronalen Netzes, als auch der Einfluss der Eingangsdaten in das neuronale Netz untersucht. Die Änderung der Netzstruktur und der Einfluss verschiedener Merkmale sind unabhängig von der Methode, wie die Merkmale in das Spracherkennungssystem integriert werden. Verschiedene Merkmale des Kurzzeit- und Langzeitspektrums stellen unterschiedliche Aspekte des Sprachsignals in den Vordergrund. Durch die Kombination verschiedener Merkmale profitiert das neuronale Netz von diesen unterschiedlichen Wissensquellen. Diese Arbeit zeigt, dass ein künstliches neuronales Netz die Informationen verschiedener Merkmale besser ausnutzen kann, als die Kombination der auf diesen Merkmalen trainierten Einzelsysteme. Die Systemkombination profitiert von den unterschiedlichen Fehlern der Einzelsysteme. Jedes Einzelsystem ist dabei nur auf einem Merkmalsset trainiert. Zusätzlich vereinfacht das neue Kombinationsverfahren die Entwicklung des finalen Spracherkennungssystems, da nicht mehrere Systeme mit unterschiedlichen Eingangsdaten trainiert werden müssen. Die so eingesparten Ressourcen können eingesetzt werden um z.B. komplexere künstliche neuronale Netze zu trainieren.

Die Struktur eines künstlichen neuronalen Netzes besitzt einen großen Einfluss auf die Qualität der erzeugten Merkmale. Diese Arbeit untersucht die Auswirkungen hierarchischer Ansätze, der Flaschenhalsarchitektur (engl. bottle-neck) und die Verwendung von rekurrenten neuronalen Netzen auf die Erkennungsleistung. Im hierarchischen Ansatz werden mehrere neuronale Netze hintereinander geschaltet, so dass als Eingang der Ausgang eines vorherigen Netzes genommen wird. Rekurrente Netze führen ein Gedächtnis ein, welches die vorherigen Eingangssignale repräsentiert. Das Hauptaugenmerk bei der Verwendung von rekurrenten Netzen liegt in der Analyse der bi-direktionalen Netzstruktur und in der Verwendung eines Lang- und Kurzeit Gedächtnisses (engl. long-short-term-memory).

Sprachenübergreifende Merkmale reduzieren den Entwicklungsaufwand eines Spracherkennungssystems. Das Aufsetzen und Trainieren neuer Systeme vereinfacht sich durch die Wiederverwendung bereits trainierter Netze. In dieser Arbeit wird die Generalisierbarkeit solcher auf einer anderen Sprache trainierten neuronalen Netze Merkmale (engl. cross-lingual features) für das Spracherkennungssystem untersucht. Insbesondere werden die Auswirkung der Netzstruktur und die Relevanz der Anzahl an Sprachdaten, die zum Trainieren des neuronalen Netzes verwendet werden, thematisiert.

Die Fehlerfunktion eines künstlichen neuronalen Netzes ist nicht konvex und das Erreichen des globalen Optimums daher nicht garantiert. In den meisten Fällen steckt die Zielfunktion in einem lokalen Optimum fest. Das Vortrainieren der Gewichte mittels un- und überwachter Lernverfahren hilft, ein besseres lokales Optimum zu finden. In dieser Arbeit werden verschiedene un- und überwachte Lernstrategien getestet und analysiert. Zusätzlich zur Vorinitialisierung der Gewichte durch *Restricted Boltzmann Machines* wird ein neues unüberwachtes Verfahren eingeführt, die *Sparse Encoder Symmetric Machines.* Das neue Verfahren zeichnet sich sowohl durch ein klares Abbruchkriterium aus, als auch durch eine direkte Optimierung der Gewichte basierend auf der Zielfunktion. Bei den Restricted Boltzmann Machines muss die eigentliche Zielfunktion approximiert werden.

Die Merkmalsgewinnungsverfahren durch neuronale Netze können nicht nur in der automatischen Spracherkennung erfolgreich angewendet werden, sondern auch in anderen Bereichen der Mustererkennung. Diese Arbeit zeigt, dass künstliche neuronale Netze in der automatischen Handschriftenerkennung und bei der automatischen Erkennung von Gebärden Verbesserungen bringen. Die Ergebnisse aus dem Bereich der Spracherkennung werden bestätigt. Das Training von gausschen Mischverteilungssystemen auf den Merkmalen des neuronalen Netzes und den Basismerkmalen ist jedoch nicht Erfolg versprechend. Erkennungssysteme, die nur auf den neuronalen Netzen Merkmalen trainiert werden, erzielen deutlich besser Erkennungsfehlerraten.

Acknowledgement

A long time ago it started, and finally here it is: my thesis. It was a long and difficult road to go, and I would like to thank a number of people who motivated me to keep going.

First, I would like to thank my supervisor Prof. Dr.-Ing. Hermann Ney who gave me the opportunity to work at the Chair of Computer Science 6 of the RWTH Aachen University on an interesting and challenging topic.

I would also like to thank PD Dr.-Ing. Björn W. Schuller for accepting to be my second supervisor, and the interest he has shown in my work and the helpful comments. Furthermore, a special thanks goes the other member of the examining board consisting of PD Dr.rer.nat. Walter Unger and Prof. Dr.-Ing. Stefan Kowalewski who jumped in on short notice.

I am very grateful to Dr.rer.nat. Ralf Schlüter for his support and the discussion we had about my research. Furthermore, I would like to thank Ph.D. Fabio Valente who introduced me to the feature extraction methods using neural networks.

A special thank goes to my former colleagues and office mates Björn, Christian, David, Patrick and Stefan and to my new office mates Martin and Simon for the great time and the interesting and helpful discussions about my work.

I would like to thank Michaela, Sabrina, David, Jan and Jens for the time they spent proof-reading my thesis.

Moreover, I would like to thank all my former and current colleagues of the Chair of Computer Science 6 in no particular order for the good moments we had during our working time and in our free time including Arne, Daniel, Gregor, Philippe, Christoph, David, Markus, Muhammad, Matthias, Saab, Pavel and Zoltán.

I am very thankful to Tara, Bhuvana and David and all other members of the IBM Thomas J. Watson Research Center for the great time, the wonderful atmosphere and for the suggested hikes during my stay in 2011.

I won't forget the time I had with the sysadmins Stefan, Kai, Tom, Oliver and Mirko to keep the computers running and maintaining the hardware and infrastructure of the institute. A special thank goes to the secretaries, Gisela, Katja, Katrin, Andrea and Stephanie whose work is certainly not acknowledged enough.

Finally, a special thank you goes to my parents, my sisters and all the other family members.

This work was partly funded by the European Union under the integrated Human Language Technologies projects TC-STAR (FP6-506738) and this work is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001-06-C-0023. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the DARPA. Finally, this work was partly realized as part of the QUAERO project, funded by OSEO, French State agency for innovation.

Outline

CHAPTER 1

Introduction

Over the last few years, the ways people interact with each other have diversified. Communication by e-mail, in chats or on social networking sites is part of many people's everyday life and the smart phone has become a constant companion. Much of this interaction takes place in written form. Nevertheless, speech has been and still is the most common and most natural way for humans to communicate.

Additionally, human-machine interactions become more and more important. In this context automatic speech recognition systems have proven to be the best choice. Yet here, automatic speech recognition is just the first step. Automatic speech recognition based systems provide information, which is necessary for different natural language processing tasks, e.g. spoken language translation or spoken language understanding.

The main goal of automatic speech recognition systems is to convert the spoken utterance from an acoustic signal (the speech) to written text (recognized words). The recognized word sequence can be further processed by a machine translation system, a dialog system or any other text based system. Depending on the given task, automatic speech recognition systems have to fulfill a large number of requirements, e.g. running close to real time, being robust to a specific type of noise and being able to recognize a huge number of different words.

Current state-of-the-art automatic speech recognition systems are based on several statistical approaches. Given a sequence of acoustic features an automatic speech recognition system recognizes the word sequence with the highest probability. The acoustic features are derived in a pre-processing step, where the speech signal is transformed by different signal analysis methods.

Usually the features are extracted by a cascade of several frequency based filters and linear transformation methods. In this work, the feature extraction is replaced and extended by artificial neural networks (ANNs), which are able to learn the feature transformations directly from the given data. The aim is the optimization of the feature extraction process as well as

the development and integration of new acoustic features to improve the performance of current state-of-the-art automatic speech recognition systems. Moreover, the methods developed in this work are general enough to be applied to other recognition tasks like optical character recognition or automatic sign language recognition.

1.1 Statistical Speech Recognition

In recent years, the statistical approach has outperformed all other approaches in speech recognition. The goal of the statistical approach is to find the sequence of words w_1^N, which maximizes the posterior probability given a sequence of acoustic observations x_1^T. According to Bayes' decision rule, the word sequence w_1^N, which maximizes the a posteriori probability, is [Bayes 63]:

$$\begin{aligned}[w_1^N]_{\text{opt}} &= \arg\max_{w_1^N} \left\{ p(w_1^N | x_1^T) \right\} \\ &= \arg\max_{w_1^N} \left\{ p(x_1^T | w_1^N) \cdot p(w_1^N) \right\}. \end{aligned} \tag{1.1}$$

Equation (1.1) defines two stochastic models. The language model $p(w_1^N)$ provides an a-priori probability of the word sequence w_1^N whereas acoustic model $p(x_1^T | w_1^N)$ denotes the probability of observing the sequence of feature vectors x_1^T given the word sequence w_1^N.

The whole statistical automatic speech recognition system consists of four major components. During the search all these four sources are combined to obtain the optimal word sequence. Figure 1.1 summarizes the interaction as well as the connection of the feature extraction, the acoustic model, the language model and the search algorithm.

- The *signal analysis* (Section 1.2) extracts acoustic features from the input speech signal. Afterwards, the sequence of acoustic features x_1^T is passed on to the speech recognizer.

- The *acoustic model* (Section 1.3) consists of statistical models for the smallest sub-word units to be distinguished by the speech recognizer, e.g. phonemes, syllables or whole words, and a pronunciation lexicon which defines the composition of an acoustic model for a given word from the sub-word units.

- The *language model* (Section 1.4) provides the *a priori* probability of a hypothesized word sequence based on the syntax, semantics and pragmatics of the language to be recognized.

- The *search algorithm* (Section 1.5) combines the acoustic model and the language model. The final hypothesis of the search is the word sequence which maximizes Equation (1.1). The full search space for continuous speech recognition, for optical character recognition and for automatic sign language recognition consists of all possible word sequences, which can be produced by a (finite) vocabulary.

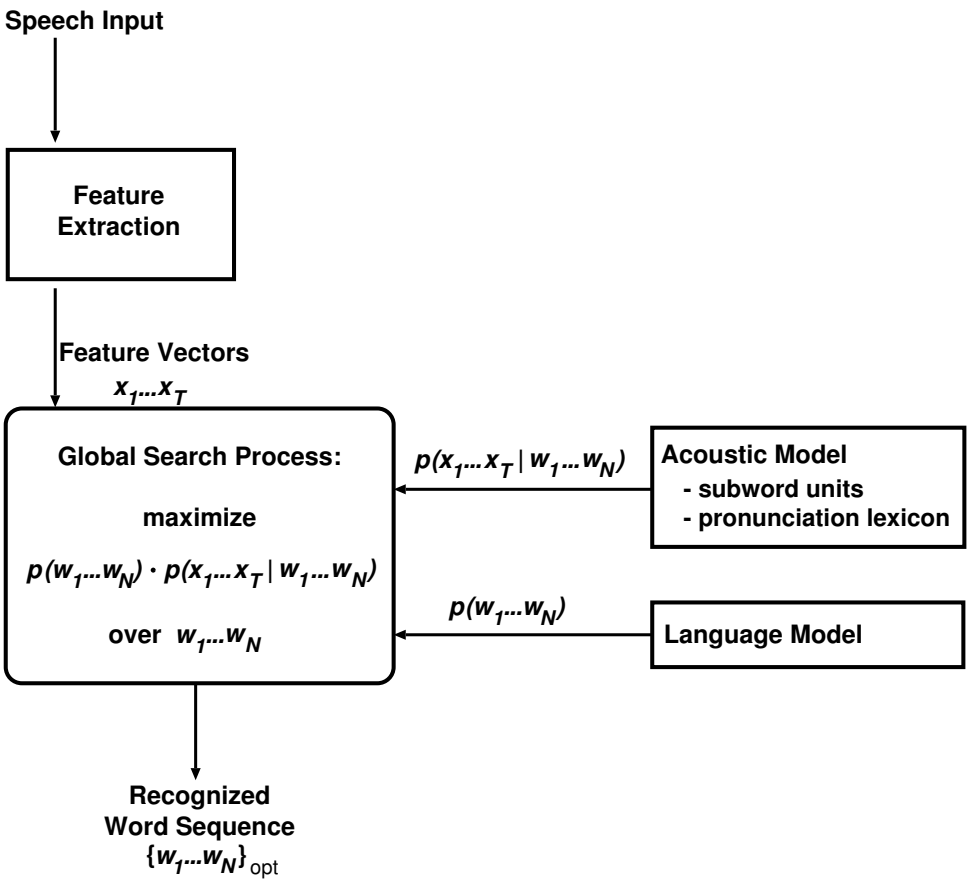


Figure 1.1 Basic architecture of a statistical automatic speech recognition system [Ney 90].

This work focuses on the signal analysis part. In the conventional feature extraction the acoustic features are obtained by the spectral analysis. Whereas the methods used in the spectral analysis are independent of the speech signal, data and task dependent feature extraction methods, e.g. classifiers like ANNs, are expected to perform better. Nevertheless, the general acoustic model training procedure is not affected by applying the data driven feature extraction methods.

1.2 Signal Analysis/ Feature Extraction

The signal analysis provides the automatic speech recognition system with a sequence of acoustic vectors. In order to obtain the best word sequence, the acoustic vectors must provide the most relevant information of the speech signal. By eliminating less significant information from the speech signal the final acoustic model becomes robust to useless and irrelevant data. Therefore, the signal analysis removes unimportant information, e.g. the intensity of the speech signal and background noise or the gender of the speaker as well as the information of the speaker's identity.

The signal analysis of current state-of-the-art automatic speech recognition systems is based on the short-term spectral analysis, usually a *fast Fourier transformation* [Rabiner & Schafer 79]. The fast Fourier transformation coefficients are further processed resulting in the *Mel frequency cepstral coefficients (MFCCs)* [Davis & Mermelstein 80] or the *perceptual linear prediction coefficients (PLPs)* [Hermansky 90]. Filtering the speech signal in the time domain by different Gammatone filters [Aertsen & Johannesma$^+$ 80], which are centered according to the Greenwood function with human parameter [Greenwood 90], yield the *Gammatone (GT)* filter based features [Schlüter & Bezrukov$^+$ 07].

All these features are motivated by the models of the human auditory system. Alongside of the acoustic features derived from the short-term power spectrum, several alternative features have been developed in the recent years, including the tandem approach [Hermansky & Ellis$^+$ 00] and features with long temporal ranges, e.g. *temporal patterns* [Hermansky & Sharma 98] or *multi-resolution RASTA* features [Hermansky & Fousek 05]. The long-term features do not perform as good as the short-term features, but provide additional information not covered by their short-term equivalents.

Recently, another feature extraction approach has been become very popular in the area of speech recognition as well as in image recognition or other related areas. In this approach the short-term or long-term features are further processed by a data driven classifier, usually an ANN, resulting in class posterior probabilities or other ANN based probabilistic features [Hermansky & Ellis$^+$ 00, Chen & Zhu$^+$ 04, Grézl & Karafiat$^+$ 07, Valente & Vepa$^+$ 07]. The further processing of short-term and long-term features and their combination by different ANNs is the main focus of this work.

Depending on the individual language recognized, specific information has to be taken into account as well. In the Chinese language for example, the tonal information plays an important role [Chen & Gopinath$^+$ 97]. Therefore, including this special information is essential to increase the performance of the acoustic model. Experimental results show that these tonal information leads to no significant improvements in European languages.

When augmenting the feature vector by its first and second derivatives, dynamic information about the speech signal is provided. When temporal context of the current frame is used, a *linear discriminant analysis* can extract the same dynamic information. In general, the linear discriminant analysis extracts better dynamic information compared to the derivative. The result of the linear discriminant analysis is a (linear) transformation, which projects a feature vector to a lower dimensional feature subspace and maximizes the class separability for distributions with equal variances. In this work the linear discriminant analysis is applied to a symmetric window of 9 or 11 adjacent feature vectors.

The feature extraction methods mentioned above are not specifically designed to be gender or speaker independent, which is, in general, hard to achieve. Nevertheless, short-term features are used to detect the gender of a speaker [Stolcke & Bratt$^+$ 00] or to identify the speaker [Doddington & Przybocki$^+$ 00] itself. Even though the information kept in the short-term features seems to be sufficient, several other methods have been developed to cope with the speaker's dependency on the acoustic features. On one hand speaker normalization techniques like *vocal tract length normalization* [Andreou & Kamm$^+$ 94, Lee & Rose 98] reduces the speaker dependency by transforming the acoustic observations. On the other hand, speaker adaptation

techniques like *maximum likelihood linear regression* [Leggetter & Woodland 95, Lee & Rose 96] adjust the acoustic model parameters to the characteristics of the given speaker. [Pitz 05] presents a comprehensive comparison of speaker normalization and speaker adaptation methods.

1.3 Acoustic Modeling

The acoustic model is a statistical model which provides the likelihood $p(x_1^T|w_1^N)$ for a sequence of acoustic features x_1^T, given a word sequence w_1^N. Instead of modeling whole words, *large vocabulary continuous speech recognition* systems use sub-word models like syllables, phonemes or phonemes including context. A pronunciation lexicon provides the mapping of a sequence of sub-word units to whole words.

Whole word models are meaningful when a small and closed vocabulary is used. Instead, smaller units allow recognizing words not covered in the training data and reducing the model complexity. They also ensure that each unit is observed in the training for reliable parameter estimation. Phonemes are the most commonly used sub-units in large vocabulary continuous speech recognition extended with one or two adjacent phonemes as context, which are called *triphones* and *quinphones*, respectively.

In natural speech a great variability in the speaker rate exists. The concept of hidden Markov models has been established in speech recognition to cope with such variations [Baker 75, Rabiner & Juang 86, Fink 03]. A hidden Markov model is a stochastic finite state automaton, represented by a number of states and the transition between these states. Whereas the observation sequence x_1^T of an hidden Markov model is visible, the state sequence s_1^T is unobservable. Therefore, the probability $p(x_1^T|w_1^N)$ is extended by some (hidden) random variables representing the states of the model:

$$\begin{aligned} p(x_1^T|w_1^N) &= \sum_{s_1^T:w_1^N} p(x_1^T, s_1^T|w_1^N) \\ &= \sum_{s_1^T:w_1^N} \prod_t^T p(x_t|x_1^{t-1}, s_1^t, w_1^N) \cdot p(s_t|x_1^{t-1}, s_1^{t-1}, w_1^N) \end{aligned} \tag{1.2}$$

Here, the sum is taken over all possible state sequences s_1^T for a given word sequence w_1^N.

Equation (1.2) can be simplified when $p(x_t|x_1^{t-1}, s_1^t, w_1^N)$ and $p(s_t|x_1^{t-1}, s_1^{t-1}, w_1^N)$ do not depend on the previous observations. Using the first order Markov assumption [Duda & Hart+ 01], the current state s_t depends on its predecessor state s_{t-1} only.

$$p(x_1^T|w_1^N) = \sum_{s_1^T:w_1^N} \prod_t^T p(x_t|s_t, w_1^N) \cdot p(s_t|s_{t-1}, w_1^N) \tag{1.3}$$

Further, the equation can be simplified by the *Viterbi* or *maximum approximation* [Ney 90]. The sum in Equation (1.3) is replaced by the maximum:

$$p(x_1^T|w_1^N) = \max_{s_1^T:w_1^N} \prod_t^T p(x_t|s_t, w_1^N) \cdot p(s_t|s_{t-1}, w_1^N) \tag{1.4}$$

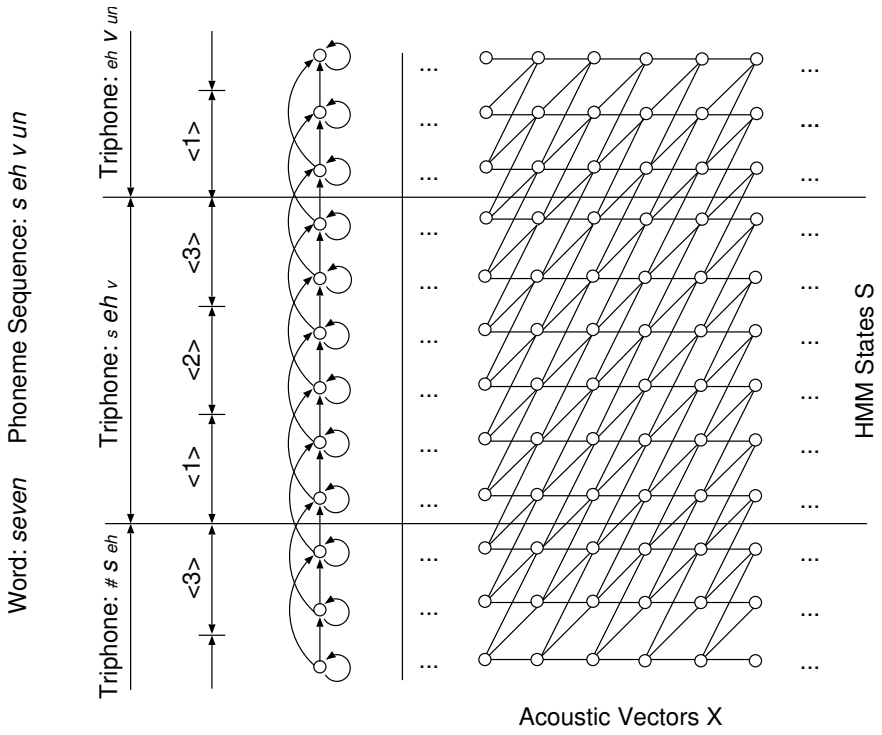


Figure 1.2 6-state hidden Markov model in Bakis topology for the triphone ${}_{s}eh_{v}$ in the word "seven" and the resulting trellis for a time alignment. The hidden Markov model segments are denoted by <1>, <2>, and <3>.

According to Equation (1.4), the probability $p(x_1^T|w_1^N)$ consists of the *emission probability* $p(x_t|s_t,w_1^N)$ and the *transition probability* $p(s_t|s_{t-1},w_1^N)$. The transition probability denotes the probability to switch from state s_{t-1} to state s_t, whereas the emission probability represents the probability to observe the feature vector x_t being in state s_t. These probabilities can be efficiently calculated using *dynamic programming* [Bellman 57, Viterbi 67, Ney 84] or the *forward-backward* algorithm [Baum 72, Rabiner & Juang 86].

Figure 1.2 illustrates an example of a hidden Markov model showing a part of the word "seven". The hidden Markov model is constructed using the Bakis topology. Next to the transitions to the next state, the Bakis model allows a self-transition and a skip transition, where one state can be skipped. In the training, the feature vectors are aligned to their corresponding state. The trellis, which is obtained by enrolling the hidden Markov model along the time axis, shows the search space for the time alignment.

In the hidden Markov model framework, the emission probabilities are modeled by discrete probabilities [Jelinek 76], by semi-continuous probabilities [Huang & Jack 89] or continuous probability distributions [Levinson & Rabiner$^+$ 83]. As commonly used, in this work the emission probabilities are modeled by the continuous probability distributions using *Gaussian hid-*

den Markov models:

$$p(x|s, w_1^N) = \sum_{l=1}^{L_s} c_{sl} \cdot \mathcal{N}(x|\mu_{sl}, \Sigma_{sl}, w_1^N), \tag{1.5}$$

where c_{sl} weights the corresponding Gaussian density $\mathcal{N}(x|\mu_{sl}, \Sigma_{sl})$ with mean vector μ_{sl} and co-variance matrix Σ_{sl}. The mixture weights have to fulfill the following constraints:

$$\sum_{l=1}^{L_s} c_{sl} = 1, \quad c_{sl} \geq 0 \tag{1.6}$$

During the training of the automatic speech recognition system, the free parameters μ_{sl}, Σ_{sl} and c_{sl} are estimated according to the *maximum likelihood* training criterion using the *Expectation-Maximization* algorithm [Dempster & Laird+ 77].

A huge number of parameters are estimated. Depending on the amount of data used, several sub-units will have few observations only. Decision tree-based state cluster algorithms (e.g. *classification and regression trees)* allow to tie these units using e.g. phonetic questions [Beulen & Welling+ 95]. The tied cluster states also provide an appropriate hidden Markov model state for unseen units.

The acoustic model improves further when the maximum likelihood trained model is re-trained using a discriminative training criterion [Schlüter 00, Povey & Woodland 02]. [Heigold 10] gives a detailed overview about different approaches using different discriminative training criterions, e.g. maximum mutual information (MMI) or minimum phoneme error (MPE).

1.4 Language Modeling

The *language model* is the third component in Figure 1.1. It covers the syntax, semantics and pragmatics of the language implicitly and provides an a priori probability of the word sequence w_1^N. In large vocabulary continuous speech recognition and other related recognition tasks, we assume that the probability of the current word w_n only depends on the $(m-1)$ predecessor words. Therefore, the resulting m-gram language model [Bahl & Jelinek+ 83] follows an $(m-1)$-th order Markov assumption. In general, the history of word w_n is a function of w_{n-m+1}^{n-1}. According to all model assumptions, the language model probability $p(w_1^N)$ is expressed as:

$$\begin{aligned} p(w_1^N) &= \prod_{n=1}^{N} p(w_n|w_1^{n-1}) \\ &\overset{\text{model assumption}}{=} \prod_{n=1}^{N} p(w_n|w_{n-m+1}^{n-1}) \end{aligned} \tag{1.7}$$

The estimation of $p(w_{n-m+1}^{n-1})$ often corresponds to the relative frequencies computed on a large training set including transcripts of speech as well as written text. It can be shown that the relative frequencies are equal to the closed form solution for $p(w|h)$, when the minimization of the *perplexity* is used as training criterion. Moreover, the performance of such a language model

is often measured according to the logarithm of the perplexity (PP) of the language model and equals the entropy of the model (Equation (1.8)).

$$\begin{aligned} \log PP &= \log \left[\prod_{n=1}^{N} p(w_n | w_{n-m+1}^{n-1}) \right]^{-1/N} \\ &= -\frac{1}{N} \cdot \sum_{n=1}^{N} \log \left[p(w_n | w_{n-m+1}^{n-1}) \right] \end{aligned} \tag{1.8}$$

When the history length m increases, the number of possible n-grams increases exponentially. Depending on the size of the vocabulary, a huge number of m-grams cannot be observed and a robust parameter estimation is not possible. This is especially problematic whenever an unobserved m-gram occurs in the testing data, as the resulting probability is zero. Therefore, the probability mass has to be distributed to unseen m-grams. Several discounting methods are known to redistribute the probability mass to all unseen events (*backing-off*) or to all events (*interpolation*) [Katz 87, Ney & Essen$^+$ 94, Generet & Ney$^+$ 95, Ney & Martin$^+$ 97]. The common approach to estimate the parameters of the smoothed language model is leaving-one-out [Ney & Essen$^+$ 94].

In the last years, neural network based language models have become very popular. The feed-forward neural networks show significant improvements over the classical n-gram LMs and are applied in an additional rescoring step [Bengio & Ducharme 01, Schwenk & Gauvain 02]. Other neural network topologies like long-short-term-memory recurrent neural networks (RNNs) [Sundermeyer & Schlüter$^+$ 12] or simpler RNNs [Mikolov & Karafiàt$^+$ 10, Mikolov & Kombrink$^+$ 11] obtain lower perplexities than the feed-forward neural networks. Due to the construction of RNNs, the language model probabilities cannot be used in a lattice rescoring step. Therefore, the RNN based language models are applied by rescoring of the n-best lists.

1.5 Search

The search module is the most important part of the speech recognizer. As shown in Figure 1.1, the search combines all knowledge sources. The goal of the search module is to find the word sequence w_1^N, which maximizes the posteriori probability $p(w_1^N | x_1^T)$ for a given feature vector sequence x_1^T. According to Equation (1.1) the best word sequence is obtained by the joint maximization of the acoustic model and the language model. If the acoustic model is a hidden Markov model and the language model an m-gram model following Equation (1.2) and Equation (1.7) respectively, the optimization problem of the search is described by:

$$\begin{aligned} [w_1^N]_{\text{opt}} &= \arg\max_{w_1^N, N} \left\{ \left[\prod_{n=1}^{N} p(w_n | w_{n-m+1}^{n-1}) \right] \cdot \left[\sum_{s_1^T : w_1^N} \prod_{t=1}^{T} p(x_t | s_t, w_1^N) \cdot p(s_t | s_{t-1}, w_1^N) \right] \right\} \\ &\overset{\text{Viterbi}}{=} \arg\max_{w_1^N, N} \left\{ \left[\prod_{n=1}^{N} p(w_n | w_{n-m+1}^{n-1}) \right] \cdot \left[\max_{s_1^T : w_1^N} \prod_{t=1}^{T} p(x_t | s_t, w_1^N) \cdot p(s_t | s_{t-1}, w_1^N) \right] \right\} \end{aligned} \tag{1.9}$$

The complexity of the optimization problem is significantly reduced by the *Viterbi* approximation as well as the *Markov* assumption. Therefore, Equation (1.9) can be efficiently solved by the dynamic programming approach [Bellman 57], dividing the whole problem into subproblems with local dependencies.

There are two main concepts to organize the search: *depth-first* or *breadth-first*. The most well-known representatives of the depth-first or stack decoding algorithm are the A^* [Jelinek 69, Paul 91] and the *Dijkstra* [Dijkstra 59] algorithm. During a depth-first search, the state hypotheses are explored in a time-asynchronous way depending on a heuristic estimate of the costs to complete the hypotheses. In contrast, in the breadth-first algorithm all hypotheses are expanded in a time-synchronous manner [Vintsyuk 71, Baker 75, Sakoe 79, Ney 84].

Due to a large vocabulary of more than 50k words in a large vocabulary continuous speech recognition system, the possible search space is huge and an exploration of the full space should be avoided. Modern speech recognizer provides several pruning techniques to reduce the search space to promising areas and to eliminate unlikely hypotheses. Due to pruning, search errors can occur when the correct hypotheses are excluded from the search space. In a depth-first decoder pruning is realized by removing non-promising hypotheses from the stack. That is why each hypothesis is evaluated due to a heuristic cost function. In the breadth-first approach the beam pruning approach is applied. In each time step only those hypotheses with their likelihood being close to the current best path are kept [Lowerre 76, Ney & Mergel$^+$ 87, Ortmanns & Ney$^+$ 97b]. However, even the Beam-search algorithm does not guarantee to find the global optimum —the global optimum can be pruned due to a poor likelihood. Nevertheless, the search space is reduced significantly. By adjusting the pruning parameters properly, the search space is reduced significantly without any search error.

On the one hand, the computational complexity of the search is easily reduced by fast likelihood computations [Ortmanns & Ney$^+$ 97a, Ramasubramansian & Paliwal 92] specialized for a single central processing unit using the *single instruction multiple data* parallelization [Kanthak & Schütz$^+$ 00], or multiple central processing units [Parihar & Schlüter$^+$ 09] or even graphic processing units [Cardinal & Dumouchel$^+$ 08]. On the other hand, several look-ahead techniques for the acoustic model or the language model [Alleva & Huang$^+$ 96, Häb-Umbach & Ney 94, Ortmanns & Ney$^+$ 96, Nolden & Ney$^+$ 11, Nolden & Schlüter$^+$ 11] reduce the computational complexity even further. The multi-pass approach reduces the search space by running a fast decoder first and applying complex methods in a re-scoring step on the reduced search space represented by a lattice, a word graph [Ljolje & Pereira$^+$ 99, Murveit & Butzberger$^+$ 93, Ney & Aubert 94, Ortmanns & Ney$^+$ 97b] or by n-best lists [Schwartz & Chow 90].

1.6 Neural Network based Feature Extraction

1.6.1 State-of-the-art: An Overview

In recent years, probabilistic features derived from an *artificial neural network (ANN)* have become a major component of current state-of-the-art speech recognizers [Hwang & Peng$^+$ 07, Chu & KuoZhang$^+$ 08, Ng & Zhang$^+$ 08, Lei & Wu$^+$ 09, Plahl & Hoffmeister$^+$ 09, Sundermeyer & Nußbaum-Thom$^+$ 11, Wöllmer & Schuller$^+$ 11]. ANNs have been used for automatic speech

recognition for the first time in the middle of the 80's [Peeling & Moore+ 86, Bourland & Wellekens 87]. There, the ANNs are used to perform full recognition of the speech, resulting in very complex ANN topologies, e.g. the *time delay neural networks* [Waibel & Hanazawa+ 89]. Nevertheless, ANN based automatic speech recognition systems have not been very successful and have been outperformed by the concept of hidden Markov models [Rabiner & Juang 86, Fink 03].

After that, two main approaches are developed to include information extracted by ANNs in the hidden Markov model decoding framework. The *hybrid* approach [Bourlard & Morgan 93] has not been very successful for automatic speech recognition unless the ANNs are trained on clustered *triphone states* or *context dependent states* [Seide & Gang+ 11, Mohamed & Sainath+ 11]. In contrast, the *tandem* approach [Hermansky & Ellis+ 00] provides an easy way to include the information from an ANN into the Gaussian hidden Markov model based recognition system. In the first years, the systems trained on ANN based features only do not achieve a competitive performance as other Gaussian hidden Markov model based systems, which are usually trained on short-term MFCC or PLP features. This changes when the structure of the ANN becomes very complex, the number of hidden layers is increased or the number of units used in the network is enlarged [Valente & Magimai-Doss+ 11].

The gain in performance of the features derived by ANNs is not limited to automatic speech recognition. In the area of image recognition, especially in optical character recognition and handwriting recognition, the hybrid and the tandem approach are applied successfully. Features derived from a *multi-layer perceptron* [Dreuw & Dötsch+ 11, Espana-Boquera & Castro-Bleda+ 11] or derived from a *recurrent neural network (RNN)* [Graves & Bunke+ 07, Graves & Schmidhuber 08, Dötsch 11] improve the recognition performance. More details about RNNs will be presented in Chapter 5. The hybrid and the tandem approach as well as the multi-layer perceptron based feature extraction approach are described in detail in Chapter 3.

Starting with [Hinton & Salakhutdinov 06] *deep belief networks* become very popular in statistical classification tasks. Due to many local optima in the optimization function of ANNs, the weight initialization of deep belief networks is critical. [Hinton & Salakhutdinov 06] provide an easy and efficient method to initialize the weights of the hidden layers using an unsupervised training strategy based on *Restricted Boltzmann Machines*. This deep network structure in combination with the pre-training of the weights is successfully transferred from the image recognition task to automatic speech recognition [Mohamed & Yu+ 10, Mohamed & Sainath+ 11]. When these *deep belief networks* are trained on clustered triphone or context dependent states the hybrid approach achieves competitive or even better recognition error rates than the corresponding Gaussian hidden Markov model based recognition system [Seide & Gang+ 11, Sainath & Kingsbury+ 11, Seide & Li+ 11, Tüske & Sundermeyer+ 12]. More details about the pre-training of the neural network weights are given in Chapter 9.

1.6.2 Neural Networks

The concept of ANNs is inspired by the neural system of the brain of mammals. The information in the brain is processed by a huge number of neurons which are connected to each other. In computer science, ANNs have been evolved as one of the models for pattern recognition

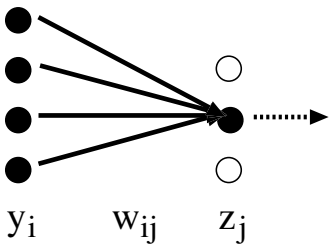


Figure 1.3 Node activation.

and machine learning tasks. An ANN is defined as a set of neurons which are linked to each other via weighted connections. A *neuron* j, also called *cell*, *unit* or *node*, consists of an *input activation* z_j and an *output activation* y_j. In the following we will use the term *node*. The input activation of a node j is the weighted sum over the output activation of all other nodes, which are connected to node j and a bias. As summarized by Figure 1.3 the neuron input activation z_j is given by:

$$z_j = \sum_i w_{ij} \cdot y_i + b_j \tag{1.10}$$

where y_i is the output activation of the neuron i and w_{ij} is the weighting factor to model the sensibility of neuron j to the activation of neuron i. The bias b_j can be encoded in the weight matrix as w_{0j}, resulting in an extended feature vector $\hat{y}_j = [1, y_j]$.

According to the weighting factor w_{ij} neuron i stimulates ($w_{ij} > 0$) or inhibits ($w_{ij} < 0$) neuron j. When no connection between neuron i and neuron j exists $w_{ij} = 0$.

To visualize the connections of an ANN the neurons are grouped and arranged in layers. A *feed-forward multi-layer perceptron* is an ANN, in which each neuron within a layer is connected to other neurons of the next layer. No backward connection to previous layers or loop connections within the same layer exist. Whereas the activation of the input layer as well as the output layer of an ANN are visible, the activation of the other layers are hidden. Hidden nodes discover regularities in the data and enrich the family of functions the network is able to approximate. The networks used in this work are limited to three hidden layers. Figure 4.1 on page 60 illustrates the general structure of such feed-forward multi-layer perceptrons.

In the mammal's brain a neuron is active and fires whenever the input activation exceeds a specific threshold. In computer science the *output activation* of a neuron is modeled by a so-called *activation function* σ (Equation (1.11)). Section 1.6.3 describes and explains different activation functions in detail.

$$y_j = \sigma(z_j) \tag{1.11}$$

Over time, the impact of each neuron changes, resulting in lower or higher weighting factors. Whereas these updates are performed all the time in the mammal's brain, in computer science a learning rule specifies how to modify the weight connections. Therefore, different examples are provided and the ANN adjusts the weights according to this training data. In the classification task, the output of the network equals 1 for the correct class and 0 anywhere else. Two typical criteria are commonly used to obtain the correct weights solving the given task: the *squared*

error criterion and the *cross-entropy error criterion*. Section 1.6.4 describes each of the training criteria and the corresponding rules to update the weight connections.

The classification tasks solved by ANNs are limited by the number of layers used. Providing an input and an output layer only the problem to be solved must be linear separable. The *XOR* problem [Zell 94, pp. 99], [Bishop 96, pp. 85] demonstrates this problem. In order to overcome this limitation, additional layers have to be included in the ANN. With one hidden layer, the ANN is able to approximate any arbitrary polynomial function. The first part of the network provides a transformation of the input features which can be solved linearly in the second part. More complex decision tasks, e.g. decision boundaries mixed into complex polygons, are solved by providing a second hidden layer.

1.6.3 Activation Function

The activation function σ defines the output activation of the neuron given the input. Figure 1.4 illustrates different activation functions which are described by Equation (1.12) to Equation (1.18) [Zell 94, pp. 77]. Most activation functions have in common that the activation function close to 0 can be modeled to be linear.

Identity/ linear:

$$y_j = z_j \tag{1.12}$$

Identity until saturation:

$$y_j = \begin{cases} -1, & z_j < -1 \\ z_j, & -1 \leq z_j \leq 1 \\ 1, & z_j > 1 \end{cases} \tag{1.13}$$

Thresholding (binary):

$$y_j = \begin{cases} -1, & z_j \leq \alpha \\ 1, & z_j > \alpha \end{cases} \tag{1.14}$$

Sinus until saturation:

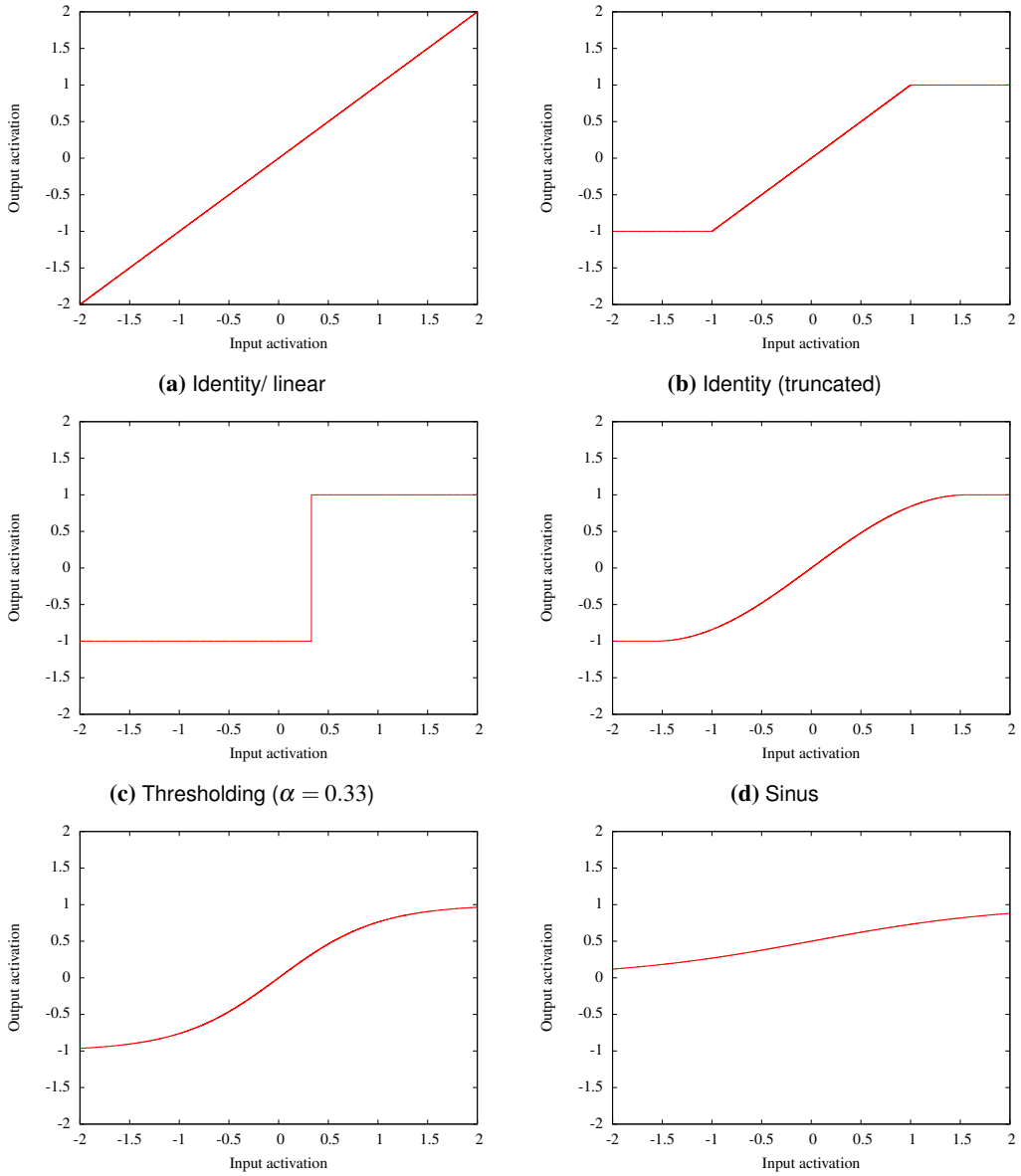
$$y_j = \begin{cases} -1, & z_j < -1 \\ \sin(z_j), & -1 \leq z_j \leq 1 \\ 1, & z_j > 1 \end{cases} \tag{1.15}$$

Hyperbolic tangent (tanh):

$$y_j = \tanh(z_j) \tag{1.16}$$

Logistic/ sigmoid:

$$y_j = \frac{1}{1+e^{-z_j}} \tag{1.17}$$



(a) Identity/ linear

(b) Identity (truncated)

(c) Thresholding ($\alpha = 0.33$)

(d) Sinus

(e) Hyperbolic tangent (tanh)

(f) Logistic/ sigmoid

Figure 1.4 Commonly used activation functions [Zell 94].

Softmax:

$$y_j = \frac{e^{z_j}}{\sum_i e^{z_i}} \tag{1.18}$$

Rectified linear:

$$y_j = \max(0, z_j) \tag{1.19}$$

In this work the sigmoid activation function (Equation (1.17)) is used for almost all neurons. The softmax activation function (Equation (1.18)) is applied in the output layer in combination with the cross-entropy error criterion. In the case of bottle-neck features the activation of the bottle-neck layer is skipped which results in the linear or identity activation (Equation (1.12)) of the corresponding node.

1.6.4 Neural Network Parameter Training

The parameters of an ANN are trained using the *back-propagation* algorithm [Rumelhart & Hinton$^+$ 86]. The training of the ANN is performed by alternating a forward and a backward step. In the forward pass the node activations in each layer are calculated, starting from the input layer. In the backward pass the derivative of the objective function w.r.t. the parameters of the network are derived. Finally, all the parameters are updated. Even though the back-propagation algorithm has been developed in context of multi-layer perceptrons, it can be applied to any feed-forward directed network.

The main goal of the back-propagation algorithm is to find a parameter configuration of the network which minimizes the global error [Zell 94, Chapter 8]. The global error E is defined as the sum over the complete training set, where each training sample (x_n, c_n) results in the local error E_n:

$$E = \sum_{n=1}^{N} E_n \tag{1.20}$$

The local error E_n is described by a specific error function. Two typical representatives of these error functions are used to train ANNs. First, the training with the *squared error criterion* (Section 1.6.4.1) will be explained, followed by the *cross-entropy error criterion* (Section 1.6.4.2). Finally, Section 1.6.4.3 describes the influence of the local error on the final weight updates $\Delta_{w_{ij}}$. The new parameters set w_{ij} of the ANN is obtained by:

$$w_{ij} \rightarrow w_{ij} + \Delta w_{ij} \tag{1.21}$$

1.6.4.1 Squared Error Criterion

Using the *squared error criterion* the local error E_n for a particular input pattern x_n is the squared difference of the output y_k obtained from the network and the reference $\hat{y}_k$

$$E_n = \frac{1}{2} \sum_{k=1}^{K} [y_k - \hat{y}_k]^2 \tag{1.22}$$

Often the network is used for classification. The output of the network corresponds to the result of the classification task and is described by the Kronecker δ function:

$$\delta(k,c) = \begin{cases} 1, & c = k, \\ 0, & c \neq k \end{cases} \tag{1.23}$$

By substituting Equation (1.23) into Equation (1.22) the local error E_n becomes:

$$E_n = \frac{1}{2} \sum_{k=1}^{K} [y_k - \delta(k, c_n)]^2 \tag{1.24}$$

In order to find the parameters minimizing the error, the gradient of the error function w.r.t. the weights is calculated. The update of the weight $w_{ij}^{(l)}$ in layer l is given by $\frac{\partial E_n}{\partial w_{ij}^{(l)}}$. E_n depends only on $w_{ij}^{(l)}$ via the input $y_i^{(l-1)}$. Applying the Chain rule we obtain:

$$\frac{\partial E_n}{\partial w_{ij}^{(l)}} = \frac{\partial E_n}{\partial z_i^{(l-1)}} \cdot \frac{\partial z_i^{(l-1)}}{\partial w_{ij}^{(l)}} \tag{1.25}$$

The second term in Equation (1.25) is simplified using Equation (1.10) to

$$\frac{\partial z_i^{(l-1)}}{\partial w_{ij}^{(l)}} = y_j^{(l-1)} \tag{1.26}$$

The first term in Equation (1.25) is often referred to as the *error*. In further equations the following short notation will be used:

$$\delta_i^{(l)} = \frac{\partial E_n}{\partial z_i^{(l-1)}} \tag{1.27}$$

By substituting Equation (1.26) and Equation (1.27) into Equation (1.25) we obtain

$$\frac{\partial E_n}{\partial w_{ij}^{(l)}} = \delta_i^{(l)} \cdot y_j^{(l-1)} \tag{1.28}$$

The error of a node in the output layer L depends on the activation function σ of this layer as well as the difference between the obtained output and the reference. The error in the last layer is obtained by

$$\delta_i^{(L)} = \sigma'(z_i^{(L)}) \cdot (y_k - \delta(k, c_n)) \tag{1.29}$$

In order to estimate the error of a node in the hidden layer we have to keep track of all connections.

$$\begin{aligned} \delta_i^{(l)} &= \frac{\partial E_n}{\partial z_i^{(l)}} = \frac{\partial E_n}{\partial y_i^{(l)}} \cdot \frac{\partial y_i^{(l)}}{\partial z_i^{(l)}} \\ &= \sigma'(z_i^{(l)}) \cdot (\sum_k w_{ki}^{(l+1)} \delta_k^{(l+1)}) \end{aligned} \tag{1.30}$$

Whereas the activation depends on the previous layer $l-1$, the error of the current layer l depends on the errors of the next layer $l+1$. The error is propagated backwards through the network until the input layer is reached. When the errors of all nodes in the network are known the final weight connection updates are performed. In the notation we skipped the layer index of the activation function σ. Since the activation function can be changed in each layer, the corresponding derivative has to be calculated.

1.6.4.2 Cross-entropy Error Criterion

Instead of the squared error criterion, the most common error criterion used for classification tasks is the cross-entropy error criterion in combination with the softmax activation function in the last layer.

When the cross-entropy error criterion is used, the local error E_n for a particular input pattern x_n changes to

$$E_n = -\sum_{k=1}^{K} \delta(k, c_n) \ln(y_k), \tag{1.31}$$

where $\delta(k,c)$ is the Kronecker function described in Equation (1.23).

As shown in the previous section, the derivatives of the local error differ in the output layer L from the error in the hidden layers $1 \leq l < L$.

Output Layer

The softmax activation function (Equation (1.18)) used in the last layer depends on all nodes. Therefore, the derivative of the output w.r.t. the node input $z_k^{(L)}$ is given by the Chain rule:

$$\frac{\partial E_n}{\partial z_k^{(L)}} = \sum_{k'=1}^{K} \frac{\partial E_n}{\partial y_{k'}^{(L)}} \cdot \frac{\partial y_{k'}^{(L)}}{\partial z_k^{(L)}} \tag{1.32}$$

The derivative of the softmax function (Equation (1.18)) results in

$$\begin{aligned} \sigma(x) &= \frac{e^x}{\sum_i e^{x_i}} \\ \sigma'(x) &= \frac{e^x}{\sum_i e^{x_i}} - \left(\frac{e^x}{\sum_i e^{x_i}} \right)^2 \\ &= \sigma(x) - \sigma(x)^2 \end{aligned} \tag{1.33}$$

Taking the derivative of the softmax function and Equation (1.23) into account, the second term in Equation (1.32) becomes

$$\frac{\partial y_{k'}^{(L)}}{\partial z_k^{(L)}} = y_k^{(L)} \delta(k,k')^{(L)} - y_k^{(L)} y_{k'}^{(L)} \tag{1.34}$$

The first term in Equation (1.32) simplifies to:

$$\frac{\partial E_n}{\partial y_k^{(L)}} = -\frac{\delta(k,c_n)}{y_k^{(L)}} \tag{1.35}$$

Substituting Equation (1.34) and Equation (1.35) back into Equation (1.32) the following equation for the final error is obtained

$$\begin{aligned}\frac{\partial E_n}{\partial z_k^{(L)}} &= -\sum_{k'=1}^{K} \frac{\delta(k',c_n)}{y_{k'}^{(L)}} \cdot \left(y_k^{(L)} \delta(k,k')^{(L)} - y_k^{(L)} y_{k'}^{(L)} \right) \\ &= y_k^{(L)} \sum_{k'=1}^{K} \delta(k',c_n) - \sum_{k'=1}^{K} \frac{y_k^{(L)}}{y_{k'}^{(L)}} \delta(k',c_n)\delta(k,k')\end{aligned} \tag{1.36}$$

Remember, we have defined the Kronecker $\delta(\cdot)$ function in such a way that we get a 1 only for the correct class and 0 everywhere else. Substituting this hard target labelling of Equation (1.23) into Equation (1.36) we obtain the same results as in Equation (1.29), where a linear activation function is used in the last layer and the squared error criterion.

$$\frac{\partial E_n}{\partial z_k^{(L)}} = y_k^{(L)} - \delta(k,c_n) \tag{1.37}$$

When the combination of the error criterion and the activation function in the last layer results in Equation (1.37), the combination of the error criterion and the activation function is called natural pairing [Bishop 96, Chapter 6] [Dunne 07, pp. 45].

[Bishop 96, Chapter 6] suggests the following natural pairings of the error function and the activation function in the last layer:

- The squared error criterion and the linear activation function
- The cross-entropy error criterion for two classes and the logistic activation function
- The cross-entropy error criterion for multiple classes and the softmax activation function

Hidden Layer

The update for the hidden layer follows Equation (1.25). The first term of Equation (1.25) describes the error $\delta_i^{(l)} = \frac{\partial E_n}{\partial z_i^{(l)}}$ of node i in the hidden layer $1 \leq l < L$. By taking the activation function into account the error is reformulated using the Chain rule to:

$$\begin{aligned}\delta_i^{(l)} &= \frac{\partial E_n}{\partial y_i^{(l)}} \cdot \frac{\partial y_i^{(l)}}{\partial z_i^{(l)}} \\ &= \frac{\partial y_i^{(l)}}{\partial z_i^{(l)}} \sum_{k=1}^{K} \frac{\partial E_n}{\partial z_k^{(l+1)}} \frac{\partial z_k^{(l+1)}}{\partial y_i^{(l)}}\end{aligned} \tag{1.38}$$

Substituting back, Equation (1.38) simplifies to

$$\delta_i^{(l)} = \sigma'(z_i^{(l)}) \sum_{k=1}^{K} \delta_k^{(l+1)} w_{ki}^{(l+1)} \tag{1.39}$$

where $\sigma'(\cdot)$ is the derivative of the activation function $\sigma(\cdot)$ which is usually the sigmoid function (see Equation (1.17) and Figure 1.4 (f)).

1.6.4.3 Weight Update and Regularization

Having estimated the output activation $y_i^{(l)}$ of each node i in layer l in the forward step and the corresponding error $\delta_i^{(l)}$ in the backward step, the weight connections $w_{ij}^{(l)}$ from layer $l-1$ to layer l are updated using the following rule:

$$\Delta w_{ij}^{(l)} = -\eta \frac{\partial E}{\partial w_{ij}} \tag{1.40}$$

where η is a constant factor also known as the *learning rate*. The new parameter set w_{ij} is obtained by substituting the correspoding update Δw_{ij} into Equation (1.21).

As shown in the previous section the cross-entropy error criterion together with the softmax activation in the last layer and the squared error criterion using the linear activation function in the output layer result in the same update rule for the weights. Including the individual errors in Equation (1.40), the following update rule is obtained:

$$\Delta w_{ij}^{(l)} = -\eta \cdot \delta_i^{(l)} y_j^{(l-1)} \tag{1.41}$$

Depending on the strategy chosen to update the weights, the weight update is either performed after just one training sample (*online learning*) or after a bunch of training samples (*batch learning*).

Since the loss function is non convex, the global optimum is not guaranteed. Often several similar solutions exist. During the ANN training the loss function can get stuck in a poor local optimum. In order to avoid such poor local optima a regularization term is added. The most common regularization terms used in the ANN training are *weight decay*, a *momentum term* or *early stopping*. [Zell 94, Chapter 9] and [Bishop 96, Chapter 7 and 9] present more details concerning the different regularization terms. In the following, we will briefly describe these three approaches.

Momentum Term

Many ANN learning algorithms contain a momentum term in the loss function [Rumelhart & Hinton$^+$ 86], [Zell 94, Chapter 9], [Bishop 96, Chapter 7 and 9]. The momentum term is a very simple technique to include a term which influences the motion through the weight space. It deals with large areas in the weight space where the convergence is slow and it avoids oscillation of the gradient. The momentum term forces the update of the gradient in the direction of the mean gradient and speeds up the convergence.

The momentum term is realized by adding the previous update to the current update of the weights. The impact of the previous weight update is controlled by a scaling factor α.

$$\Delta w_{ij}(t) = -\eta \frac{\partial E_n}{\partial w_{ij}} + \alpha \Delta w_{ij}(t-1) \tag{1.42}$$

Weight Decay

Another possibility to introduce a regularization term in the loss function is weight decay [Bishop 96, pp. 338], [Zell 94, pp. 117].

$$\tilde{E}_n = E_n + \frac{\lambda}{2} \sum_{i,j} w_{ij}^2 \tag{1.43}$$

$$\begin{aligned} \Delta \tilde{w}_{ij}(t) &= -\eta \frac{\partial \tilde{E}_n}{\partial w_{ij}} \\ &= -\eta \frac{\partial E_n}{\partial w_{ij}} + \underbrace{\eta \lambda}_{\tilde{\lambda}} w_{ij} \end{aligned} \tag{1.44}$$

Weight decay penalizes large weight values which are responsible for an over-fitted mapping with regions of large curvature. For small weight values the network mapping represented by a multi-layer perceptron is approximately linear, since the central region of the sigmoid or other activation function can be approximated by a linear transformation.

Early Stopping

One main motivation to introduce a regularization term in the loss function is to avoid over-fitting to the data. The early stopping criterion [Bishop 96, pp. 343] is an alternative method to add regularization to the loss function. The performance of the training is measured on an independent validation set. A decrease on the validation set in an early stage of the ANN training shows a poor generalization of the trained network. Therefore, the training can be stopped at the point with the smallest error w.r.t. the validation set, since the current network configuration is expected to give the best performance to unknown data.

CHAPTER 2

Scientific Goals

Features based on ANNs have become a major component of current state-of-the-art speech and image recognizers. Whereas the training of large and complex ANNs is extremely time consuming and requires a huge amount of computational resources, the forwarding step of an ANN is fast. The forwarding step contains mainly matrix multiplications which can be efficiently performed. Due to the construction of the ANN and the softmax activation function for the output layer class posterior estimates and other probabilistic features can be derived by an ANN.

The class posterior estimates of an ANN are used in two different ways in the recognition system. In the *hybrid* approach the class posterior probabilities of the ANN are used as state emission probabilities for a Gaussian hidden Markov model based recognizer. Within this framework no Gaussian Mixture estimations are required. Nevertheless, adaptation techniques like speaker adaptive training and discriminative training could not be applied. In the *tandem* approach the posteriors or probabilistic features are taken as input to train a Gaussian hidden Markov model based recognizer. The ANN features can either be used as the only feature stream or as additional feature stream.

On the one hand the objective of this thesis is to compare the two ways to include the class posterior probabilities into a Gaussian hidden Markov model based recognition system. On the other hand, new topologies and ANN based features for several languages are recommended. In this work, the methods are evaluated on different speech, image and sign language recognition tasks.

Comparison of Input Features and Training Classes (Chapter 3)

In recent years, a huge number of new feature extraction methods have been published for speech and image recognition. In automatic speech recognition each of these feature sets handles a specific type of the speech production system or of the recording environment, e.g. to suppress different types of noise in the audio signal or to model long-term dependencies of the speech signal. Whereas all of these features improve the system performance for the given task, normally one short-term feature stream is used to train a Gaussian hidden Markov model based speech recognition system.

This thesis covers several short-term and long-term feature sets and evaluates which feature sets should be used as input to train the ANN as well as how the final ANN based features should be included in the recognition system to obtain optimal performance.

Gaussian hidden Markov model based speech recognition systems are mostly trained on classes of context dependent states, created by a cluster algorithm (e.g. classification and regression tree) from triphones, quinphones or septphones. In contrast, ANNs are mostly trained on phoneme or phoneme states of the Gaussian hidden Markov model. Our aim is to analyze the different classes for the ANN training and to evaluate the effect of these classes w.r.t. the overall performance of the automatic speech recognition system. Therefore, we analyze the tandem and hybrid approach on the different ANN class posteriors.

Investigation of the Structure and Topology of neural networks (Chapter 4 & Chapter 5)

From the theoretical point of view a feed-forward ANN consisting of one hidden layer can approximate any arbitrary function. The precision of the approximation is given by the size of the hidden layer. In praxis, a huge hidden layer size is not trainable and multiple hidden layers are introduced instead. Nevertheless, the information presented at each hidden layer of a feed-forward ANN is limited to the output of each time step of the previous layer. Providing additional contextual information in a second ANN helps to avoid this limitation. The output of a previous trained ANN including temporal information is presented as input. Moreover, additional features are used as input to further improve the recognition performance. In contrast, the temporal context can be provided by recurrent connections where the activations are looped back to the input. We analyze the structure of RNNs and their performance compared to the non-recurrent networks.

The training of the network is performed in a supervised manner and therefore the posteriors of an ANN are limited by the alignment given for training. Instead of using these posteriors of the network, the activations of an inner layer could be taken into account. Due to the indirect connections of these inner layers to the final output, these features can be seen as an abstract or intermediate representation of the ANN posteriors. This work addresses different possibilities to create such features and discusses the advantages as well as the disadvantages of these two approaches. Instead of including the temporal information in the features, the RNNs are able to handle these temporal dependencies due to their construction.

Inspired by this baseline structure, several new complex structures are developed and their impact for the different systems and tasks are determined.

Portability of neural network Features (Chapter 6).

A simple ANN consists of an input and an output layer and one hidden layer. This ANN structure is outperformed by complex structures, where multiple ANNs are trained or the output of an inner layer is taken. The training time of such complex topologies increases with the complexity of the network.

The common approach to provide features to a new task is to perform a full retraining of the ANN starting from scratch. In order to save computational resources reusing previously trained ANNs and porting to new tasks and domains is helpful. This work explores the portability of such trained ANNs for two European languages as well as the degree of kinship of the language and the structure of the ANN.

Investigations on Feature Combination (Chapter 7)

A large variety of input features for automatic speech recognition as well as for image recognition exists. Classical feature combination techniques like concatenation or linear discriminant analysis are suboptimal to cover all the information presented in the combined feature stream. In recent years, system combination seems to be superior to other combination approaches. Therefore, individual systems based on each feature type are trained and the results are combined by *ROVER* or lattice based confusion networks. Developing a huge number of individual systems requires a huge number of available resources. Combining the information of several feature streams in an earlier stage of the system development cycle is one of the important challenges to be addressed.

In this work, feature combination techniques based on ANNs are recommended. Therefore, several simple and complex combination techniques are investigated where even the simple techniques outperform the system combination techniques.

Parameter Scaling (Chapter 8)

The available amount of data to train the recognition systems increases continuously. This work investigates the effect of the number of parameters in the network according to the amount of data used.

Supervised and Unsupervised Weight Initialization (Chapter 9)

Depending on the number of hidden layers in an ANN, the training of such an ANN is a hard challenge. Finding a good initialization of the weights to perform the network training helps to solve the problem of training ANNs with more than 4 hidden layers get stuck in local optima. The initial weights of this deep neural network can be trained by the concept of Restricted Boltzmann Machines. In order to update the weights of the deep neural network the individual weight connections of the layers are trained by Restricted Boltzmann Machines. The Restricted Boltzmann Machines require the concept of contrastive divergence to update the weight connections.

To overcome this problem, this thesis establishes an alternative method based directly on the loss function. Moreover, the sparseness of the output of the ANN can be directly controlled and is included in the optimization process. A detailed comparison of the two unsupervised training method are given. Additionally, this thesis addresses the open question if the unsupervised initialization is necessary or if supervised training techniques are sufficient to initialize the weights. The analyses are performed on two topologies, the conventional single ANN and the bottle-neck approach which are explained in detail in Chapter 4.

Generalization of ANN based features (Chapter 10)

All previously shown experiments cover examples taken from the speech recognition task. The concept of ANN based features is not limited to speech recognition and can be applied to other tasks as well. This work shows how ANN based features can be used in several optical character recognition tasks as well as for automatic sign language recognition. The ANN based features obtain the best performance results currently achieved on these image corpora.

CHAPTER 3

Input Features and Target Classes for Neural Network Training

In this chapter we analyze and discuss the integration of probabilistic features derived from an ANN into state-of-the-art automatic speech recognition systems. A huge number of different methods exist to pre-process the input features. The significance of the input features used, as well as the preprocessing of these features for ANN training is investigated. The integration includes several feature adaptation as well as several feature reduction techniques. In addition, we analyze three different target classes for ANN training.

The chapter is structured as follows: First, the hybrid and the tandem approach will be analyzed in Section 3.1. Currently, these two methods are known to integrate the posterior estimates or probabilistic features derived from an ANN into state-of-the-art automatic speech recognition systems. Therefore, the ANN trainings are performed on phonemes, phoneme states and triphones, context dependent states or the corresponding classification and regression tree labels. In Section 3.2 the integration of the probabilistic features into the tandem based system is optimized. Finally, we investigate the relevance of different feature adaptations and pre-processings to obtain best system performance in Section 3.4. The feature adaptations include vocal tract length normalization and constrained maximum likelihood linear regression for short-term and long-term features.

3.1 Integration of Artificial Neural Network based Posterior Features

Already in the 1980's ANNs have been used for automatic speech recognition [Peeling & Moore+ 86, Bourland & Wellekens 87]. In the beginning, several different problems occur, resulting in very complex ANN topology like the *time delay neural networks* [Waibel & Hanazawa+ 89]. Nevertheless, ANN based systems have been outperformed by the concept of hidden Markov models [Rabiner & Juang 86, Fink 03].

Starting in the late 1980's [Lippmann 89], two approaches to include the information provided by ANNs into the hidden Markov model decoding framework are developed. In the area of automatic speech recognition the *hybrid* approach [Bourlard & Morgan 93] does only achieve competitive results to current state-of-the-art hidden Markov model based systems when the network is trained on context dependent states [Mohamed & Sainath$^+$ 11, Seide & Gang$^+$ 11]. Until 2011, the *tandem* approach [Hermansky & Ellis$^+$ 00] has been the only way to successfully include the information provided by ANNs.

Section 3.1.1 and Section 3.1.2 describe the hybrid and the tandem approaches respectively. Moreover, their behavior w.r.t. the target classes used for ANN training is investigated including a comparison of the main advantages as well as the main disadvantages. Section 3.1.3 presents experimental hybrid recognition and tandem based recognition results trained on different phonetic targets. Finally, Section 3.1.4 compares the two approaches and discusses the advantages as well as the disadvantages w.r.t. the given task.

3.1.1 Hybrid Approach

The hybrid approach [Bourlard & Morgan 93] combines the decoding structure of a hidden Markov model and the posterior information provided by the ANN. In current state-of-the-art automatic speech recognition systems the observation probabilities of the hidden Markov model are the state emission probabilities $p(x|s)$. $p(x|s)$ models the probability to observe the feature vector x given the current state s. Yet, the ANN estimates the posterior probability $p(s|x)$ of observing the state or label s given the input vector x. According to Bayes' rule, Equation (3.1) describes the connection between $p(x|s)$ and $p(s|x)$:

$$p(x|s) = \frac{p(s|x) \cdot p(x)}{p(s)}, \tag{3.1}$$

where $p(s)$ is the state prior and $p(x)$ the feature prior.

Whereas the state prior $p(s)$ is taken from a previously trained hidden Markov model or derived from the relative frequencies of assigned observations in the hidden Markov model, the feature prior $p(x)$ cannot be directly derived from the model. Discarding $p(x)$ results in unnormalized scaled likelihoods.

During decoding the time distortion penalties of the hidden Markov model have to be tuned. In order to avoid adjusting the time distortion penalties of the hidden Markov model directly, scaling the state priors $p(s)$ leads to Equation (3.2) and Equation (3.3).

$$p(x|s) \approx \frac{p(s|x)}{p(s)^\alpha} \tag{3.2}$$

or

$$-\log p(x|s) \approx -\log p(s|x) + \alpha \cdot p(s) \tag{3.3}$$

when optimizing the negative log likelihood score.

Recently, the hybrid recognition approach has been re-developed for automatic speech recognition using context dependent states [Mohamed & Yu+ 10, Seide & Gang+ 11, Sainath & Kingsbury+ 11] in combination with deep belief networks. The corresponding labels for training the ANN on context dependent states are taken from a given alignment which —in our case— is obtained from a previously trained Gaussian hidden Markov model based acoustic model. The technique of pre-training the weights of an ANN provides a better initialization of the weight connections as just using randomized values. Chapter 9 describes the concept of pre-training the weights of an ANN and explains different pre-training concepts like Restricted Boltzmann Machines [Hinton & Osindero+ 06] in more detail.

3.1.2 Tandem Approach

Instead of using the posterior estimates of the ANN directly, in the tandem approach the ANN probabilistic features are provided as input to (re-)train a Gaussian hidden Markov model based system [Hermansky & Ellis+ 00]. As described in Equation (3.4), the feature can be transformed by any arbitrary feature transformation function Φ.

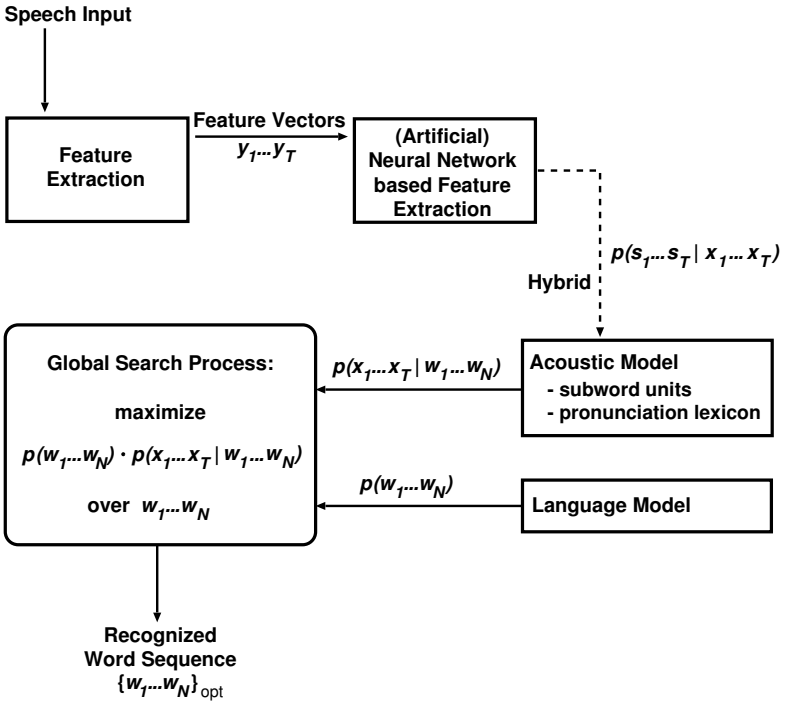
$$x'_t = \Phi(\log p(s_t|x_t)) \tag{3.4}$$

Therefore, the ANN based features are transformed by logarithm. This step is necessary to Gaussianize the features for the Gaussian hidden Markov model training. In a second important step, the features are decorrelated by discrete cosine transform, principal component analysis or linear discriminant analysis. Section 3.2 gives a comparison of the different decorrelation methods. Adding the ANN based feature extraction to Figure 1.1 results in the modified automatic speech recognition architecture shown in Figure 3.1.

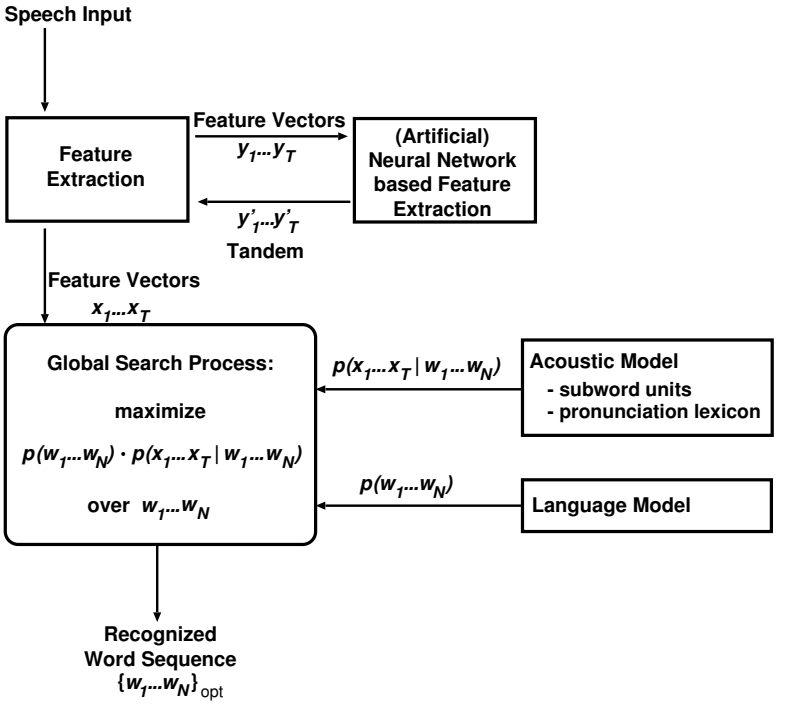
As shown in Figure 3.1 one main advantage of the tandem approach is that only the feature extraction part needs modifications. All the other parts of the training and decoding process of the Gaussian hidden Markov model based automatic speech recognition system stay unchanged. Moreover, model adaptation techniques like speaker adaptive training or discriminative training can be applied as well. Nevertheless, a full new acoustic training has to be performed on the new input features including all adaptation steps as well.

Another main advantage of the tandem approach is that a fixed size of the probabilistic features is not required. Instead of using the output activations of the ANN any activation from any inner or outer layer of the network can be used as input for the Gaussian hidden Markov model training. The number of features can be enlarged, as well as projected to a lower dimensional feature space. In speech recognition typical representatives of this concept are the hidden activation temporal patterns [Chen & Chang+ 03] and the bottle-neck features [Grézl & Karafiat+ 07]. Experimental results for these bottle-neck features are given in Section 4.3.

Several methods to combine different feature sets exists. Section 3.2 analyzes and compares some of these methods.



(a) Hybrid approach



(b) Tandem approach

Figure 3.1 Illustrations of the ANN feature integration approaches into an automatic speech recognition system using (a) the hybrid or (b) the tandem approach.

Table 3.1 Effect of different target classes on the multi-layer perceptron training frame accuracy on Quaero French. The target classes of the 2-layer multi-layer perceptron are phonemes, phoneme states or triphone states respectively. The triphone states are clustered using the classification and regression tree algorithm and silence is modeled by a single output class.

Target class	# of Classes	Training Epochs	Training frame accuracy ([%])	
			Training set	Validation set
Phonemes	44	6	66.98	66.52
Phoneme states	130	6	57.54	56.96
Triphone states	4501	9	47.50	43.38

3.1.3 Experimental Results

This section gives a systematical experimental comparison of the hybrid and the tandem approaches. The results for each approach trained on different phonetic targets are shown as well as a comparison of the two approaches afterwards. As in all other experiments, the silence state is modeled without context information.

The experimental results are performed on the French language using the Quaero database. Section A.2 gives a detailed description of the French corpora as well as additional information to the systems trained on French. The Gaussian hidden Markov model based baseline system contains a speaker independent and a speaker adapted acoustic model.

The feed-forward multi-layer perceptron networks are trained on phonemes, phoneme states or triphone states. A classification and regression tree organizes the mapping of the triphones to the corresponding triphone states. The classification and regression tree is constructed by asking specific phonetic questions [Beulen 99]. A framewise alignment provides the corresponding labels for each frame. The alignment has been created using a previously trained Gaussian hidden Markov model based acoustic model which uses the classification and regression tree of the speaker independent acoustic model.

The structure of the multi-layer perceptron is kept simple. The network contains one hidden layer with 4500 nodes. The output size is 44, 130 and 4501 for phonemes, phoneme states and triphone states respectively. The multi-layer perceptron are fed with the short-term MFCC features, its first derivative (Δ) and the second derivative of the first component ($\Delta\Delta_1$) as input. Table 3.1 summarizes the frame accuracy rate for the training of the multi-layer perceptrons. The different numbers of training classes makes a direct comparison impossible. Nevertheless, the frame accuracy rate drops when the output classes are increased.

3.1.3.1 Hybrid Recognition Results

As shown in Table 3.2, the performance on the ANN posterior estimates depends on the target class used. Hybrid recognitions on multi-layer perceptron features trained on phoneme or phoneme state level do not achieve similar recognition performance as the Gaussian hidden Markov model baseline system. The word error rate of the two systems increases by about 20-40% relative.

Table 3.2 Comparison of hybrid and Gaussian hidden Markov model based recognition systems on Quaero French. The 2-layer multi-layer perceptrons differ only in the number of target classes, trained on phonemes, phoneme states or (clustered) triphone states. MFCCs are used as input for the multi-layer perceptrons and the Gaussian hidden Markov models which is speaker adapted using SAT/CMLLR. Each system is optimized independently on the development set, marked by *.

		Total # of Parameters	Testing corpora (WER [%])			
			dev10*	eval10	eval09	dev09
GHMM	MFCC	50.0M	25.8	27.6	36.6	41.6
	+ SAT/CMLLR		24.1	25.4	33.2	38.8
Hybrid	Phonemes	1.4M	37.3	39.3	48.2	53.1
	Phoneme states	1.7M	34.8	36.3	45.1	50.4
	Triphone states	19.2M	24.6	25.3	35.4	40.2

Instead, the multi-layer perceptron trained on triphone states achieves better word error rate than the speaker independent Gaussian hidden Markov model baseline system using much less parameters. The relative improvement starts around 3% on dev09 and eval09 and scales up to 8% on eval10. However, the Gaussian hidden Markov model baseline system could be improved by speaker adaptive training, whereas no successful speaker adaptation methods for the hybrid approach are known. Currently, speaker adaptation is performed by providing speaker adapted input features. Therefore, a speaker adapted baseline system has to be trained to provide the transformation matrices (see Section 3.4.2).

Overall, hybrid recognitions outperform Gaussian hidden Markov model based systems when the ANNs are trained on the same input features and the output of the networks corresponds to the triphone states or context dependent states. No retraining of a complete system is necessary and only the decoding has to be performed. Yet, the possibility to improve the hybrid system is limited, whereas the Gaussian hidden Markov model based system can be improved by speaker adaptive training and discriminative training.

3.1.3.2 Tandem Recognition Results

Several systems are trained using the tandem approach as described above. The systems trained are based on a single feature stream or on the combination of the short-term MFCCs and the posterior estimates derived from a multi-layer perceptron. The multi-layer perceptron is again trained on MFCCs including the first derivative and the second derivatives of the first component. Three systems are trained on MFCCs on phoneme posteriors or on phoneme state posteriors. Since a huge amount of storage is required to save the context dependent state posteriors, the training on context dependent state posteriors is skipped. All features within a sliding window of size 9 undergo a linear discriminant analysis and are projected to a 45 dimensional feature space. Table 3.3 shows the tandem recognition results for the French data base.

Finally, the 90 dimensional feature vectors contain the augmented linear discriminant analysis transformed posterior and linear discriminant analysis transformed MFCC feature streams. The training of the speaker independent and speaker adapted systems are performed on the same corpus.

Table 3.3 Comparison of tandem based recognition systems on Quaero French after speaker adaptation using SAT/CMLLR. The 2-layer multi-layer perceptrons differ only in the number of target classes and are trained on phonemes or phoneme states and MFCCs as input. The acoustic models are trained on MFCCs, the multi-layer perceptron based posterior estimates or on the augmented feature stream. A linear discriminant analysis transforms each feature stream to 45 components, including a temporal context of ± 4 frames. The systems are optimized independently on the dev2010 set.

	GHMM	Testing corpora (WER [%])			
Feature type	Input size	dev10	eval10	eval09	dev09
MFCC	45	24.1	25.4	33.2	38.8
MLP (phonemes)		24.1	25.1	34.5	39.8
MLP (phoneme states)		25.5	26.7	35.5	41.2
MFCC					
+ MLP (phonemes)	90	22.9	23.8	32.8	38.2
+ MLP (phoneme states)		24.0	24.4	33.4	39.2

As shown in Table 3.3, the speaker adapted system trained on phoneme posteriors is competitive to the baseline system on the development and evaluation data of 2010 whereas the system based on phoneme state posteriors does not perform as well. Due to a higher target class size, the final feature size has to be increased as well. To compare the systems trained, the parameters of the systems and the input feature size are fixed.

The transformed short-term MFCC features augmented by the posteriors achieve the best performance. This combination improves the word error rate of the baseline MFCC system by about 5-6% relative on the development and evaluation of 2010 and a little bit less on the testing corpora of 2009.

In order to verify the results on French, the same experiments have been performed on Spanish and Chinese. Final recognition results can be found in the corresponding sections.

3.1.3.3 Hybrid and Tandem Comparison

As shown in the previous two sections, the hybrid and the tandem approach are two possibilities to include ANN posterior estimates into a Gaussian hidden Markov model based recognition system. Table 3.4 summarizes the best hybrid and tandem results using the settings described above. The tandem approach obtains the best recognition performance and the corresponding system has been trained on the MFCCs and the multi-layer perceptron phoneme posterior estimates.

If we just take into account the performance of the systems trained on the MFCC based features, the tandem system performs best. [Tüske & Sundermeyer$^+$ 12] shows corresponding results using the same feature set for tandem and hybrid recognitions.

In order to achieve the best hybrid and tandem recognition performance, the setups are modified. The topology of the multi-layer perceptron is changed as well as the input features and the number of hidden layers. Chapter 4 explains all different bottle-neck topologies of this experiment in detail. A hierarchical multi-layer perceptron is trained, where the first network uses the bottle-neck concept. In addition, the MFCCs are transformed by vocal tract length nor-

Table 3.4 Comparison of hybrid and tandem based recognition systems using multi-layer perceptron posteriors on Quaero French. The tandem systems includes speaker adaptation using SAT/CMLLR. The 2-layer multi-layer perceptrons are trained on the MFCC features and differ only in the number of target classes, phonemes and (clustered) triphone states. The phoneme posteriors are used to train a tandem system, the hybrid system is based on the triphone state posteriors. The acoustic models of the tandem systems are trained on MFCCs augmented by the multi-layer perceptron based phoneme posteriors. A linear discriminant analysis transforms each feature stream to 45 components, including a context of ± 4 frames. The systems are optimized independently on the dev2010 set.

		GHMM	Testing corpora (WER [%])			
		Input size	dev10	eval10	eval09	dev09
Tandem	MFCC	45	25.8	27.6	36.6	41.6
	+ SAT/CMLLR		24.1	25.4	33.2	38.8
	+ MLP-posteriors	90	23.6	24.9	35.0	39.6
	+ SAT/CMLLR		22.9	23.8	32.8	38.2
Hybrid	Triphone states	—	24.6	25.3	35.4	40.2

Table 3.5 Comparison of hybrid and tandem based recognition systems using bottle-neck based features as input. The tandem and hybrid systems are trained on speaker adapted features using constrained maximum likelihood linear regression, including 45 dimensional MFCCs, transformed by linear discriminant analysis and vocal tract length normalization and 62 multi-layer perceptron based bottle-neck features. Each system is optimized independently on the dev2010 set.

	Total # of	Testing corpora (WER [%])	
	Parameters	dev10*	eval10
Tandem	118M	21.6	22.7
Hybrid	33M	21.4	22.7

malization. The bottle-neck size is set to 62 components and the other hidden layers to 7000. speaker adaptive training using constrained maximum likelihood linear regression transforms the final bottle-neck features augmented by the MFCCs. The second multi-layer perceptron is trained on these speaker adapted features as well as the tandem system. The output layer for all multi-layer perceptrons corresponds to the 4501 classification and regression tree labels and the six hidden layers contain 2000 nodes each. Table 3.5 summarizes the results. In the best hybrid and tandem systems the differences between the two approaches vanish.

In order to judge the tandem and the hybrid approach, the main advantages and disadvantages of the two approaches have to be considered as well. Instead of using a previously trained system to perform the recognition, a separate training step is required in the tandem approach. If training a new speaker independent and speaker adapted acoustic model is not an issue, the tandem system will be the best choice. Since in addition to the speaker adapted system, discriminative training can be used as well to improve the tandem system further. One additional advantage of this approach is that the training of the multi-layer perceptron is faster due to a lower dimensional output layer. Another positive aspect of the tandem approach is that the dimension is not fixed and the output of an inner layer is easy to use. Feature adaptation tech-

niques do not yet exist for the hybrid approach. Nevertheless, if fast decoding is an issue and an easy system development is important the hybrid approach is the best choice.

3.1.4 Summary

This section introduced two different possibilities to integrate the posterior estimates derived from an ANN into a state-of-the-art automatic speech recognition system. Depending on the labels used in training, the hybrid and the tandem approach outperformed the baseline system, but achieved competitive performance to each other.

Hybrid recognitions achieved competitive performance when the ANNs were trained on context dependent state or triphone states. Since the output layer of the multi-layer perceptron was huge, the training needed a lot of computational resources. Such trainings were efficiently performed using graphic processing units instead of central processing units. The reader should keep in mind that there were no known adaptation techniques to be used during the multi-layer perceptron training. The adaptation information was encoded into the feature vector. Additionally, hybrid recognitions were very fast and efficient —no Gaussian computations had to be performed. Therefore, hybrid recognitions should be applied when retraining of the acoustic model is not possible and multi-pass systems are not required.

In all other cases, the tandem system will be the best choice. Nowadays, high computational power is available for a small amount of money and high computational techniques become affordable for everyone.

The tandem approach will not be limited to the use of posterior estimates. Since the number of features was not fixed, many other preprocessing steps can be applied as well. In addition to the improved feature extraction methods several efficient and optimized adaptation techniques were available for Gaussian hidden Markov model based recognition systems. These techniques were applied without any further development to the tandem systems.

Overall, the tandem approach had a lot of benefits compared to the hybrid recognition approach and seemed to be more promising than the hybrid approach. Since the tandem approach was superior, in almost all further experiments in this thesis ANN based features and the Gaussian hidden Markov model decoding structure will be combined using the tandem approach.

3.2 Optimization of the Tandem Approach

As shown in the previous section, the tandem approach is an easy and efficient concept to integrate probabilistic features derived from an ANN into state-of-the-art automatic speech recognition systems. In order to achieve the best performance, different information sources, feature combination and system combination techniques have to be applied. System combination techniques like confusion network combination as described in [Hoffmeister 11] outperform the feature combination methods [Zolnay & Schlüter$^+$ 05, Zolnay 06].

As described in [Hermansky & Ellis$^+$ 00], the most common way to integrate the multi-layer perceptron posterior features using the tandem approach is to transform and reduce the posteriors by principal component analysis. The final feature size is selected to keep at least 95% of the variability of the features.

In this section we analyze the effect of the principal component analysis and other feature combination methods. Moreover, state-of-the-art system combination methods like ROVER or confusion network combination will also be taken into account [Fiscus 97, Hoffmeister 11]. The system combination methods tested in this work are based on the recognition output, on word graphs or lattices. The main disadvantage of system combination approaches is that multiple trainings have to be performed in parallel before the outputs of the single systems can be combined.

Concatenation is the simplest feature combination method. The other feature combination methods analyzed in this thesis reduce the final feature size by principal component analysis and linear discriminant analysis. In order to benefit from temporal context, several consecutive frames are taken into account before combining and transforming the features.

The experimental results described in the following section are performed on Spanish. Section A.3 describes the Spanish Quaero corpus in detail.

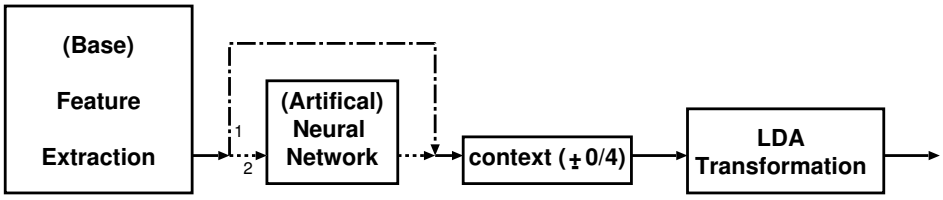
3.2.1 Experimental Results

In order to optimize the tandem approach and to achieve the optimal performance, two main experiments are performed on the Spanish Quaero corpus. The first experiments analyze the effect of system combination and feature combination methods using short-term MFCC features and the posterior features derived from an ANN. The second experiment focuses on the question of how to combine the MFCC features and the posterior estimates in the best manner.

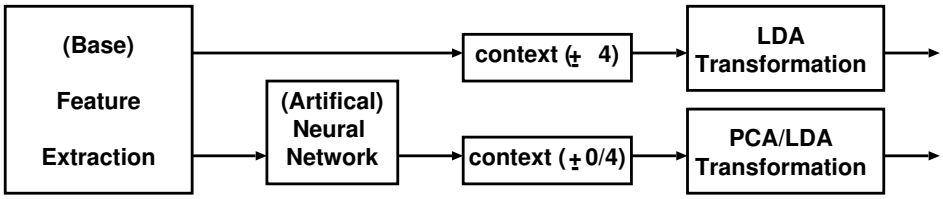
3.2.1.1 Feature Stacking vs. System Combination

Three single systems are trained in the first experiments. The first two systems are trained on the MFCCs or multi-layer perceptron based posterior estimates respectively. In order to include temporal context the features set augment 9 consecutive frames. A linear discriminant analysis reduces each expanded feature vector to 45 components. Combining the two 45 dimensional feature sets results in the 90 dimensional feature set of the third system. Figure 3.2 (a) and Figure 3.2 (b) illustrate the feature extraction of these three systems.

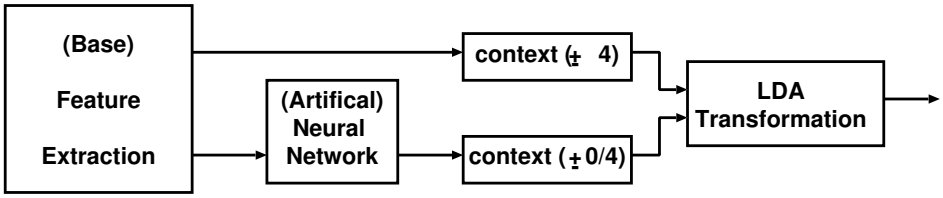
We perform the system combination experiments using several confusion network combination methods of the individual systems [Hoffmeister 11, Evermann & Woodland 00]. The combination using ROVER [Fiscus 97] results in slightly worse results compared to confusion network combinations. Table 3.6 summarizes the final feature and system combination results. As shown, each of the systems trained on a single feature stream performs equally well. Even though the performance of the systems differs around 0.4% absolute in word error rate on the development set, the difference is less on the evaluation sets. Moreover, combining the individual linear discriminant analysis transformed feature streams results in more than 1% absolute reduction w.r.t. word error rate on all test sets. This is a relative improvement of more than 5%. The worst of the three systems is improved by more than 2% absolute in word error rate. In contrast however, system combination obtains a relative improvement of 4% only. The difference is about 0.2% absolute in word error rate on dev10 and on the evaluation corpora.



(a) Single feature stream



(b) Multiple feature stream combination with multiple transformations



(c) Multiple feature stream combination with global transformation

Figure 3.2 Illustration of several feature combination architectures. In (a) the multi-layer perceptron processing is optional (path 1 or path 2) depending on the feature sets used. In (b) each feature stream is processed independently and in (c) one global transformation matrix is applied. Each feature stream includes temporal context of ± 0 or ± 4 consecutive frames.

Table 3.6 Comparison of tandem feature combination and system combination after speaker adaptation using SAT/CMLLR on Quaero Spanish. The multi-layer perceptron based posteriors are trained on the MFCCs and the 33 phoneme classes. Feature concatenation after linear discriminant analysis reduction is marked by $+$ whereas the system combination result is marked by $\oplus$. The parameters are tuned on the dev10 corpus.

	Testing corpora (WER [%])			
Feature type	dev10*	eval10	eval09	dev09
MFCC	21.6	18.2	16.7	29.8
MLP-posteriors	22.4	18.5	17.1	30.7
MFCC + MLP-posteriors	20.4	16.9	15.5	28.4
MFCC $\oplus$ MLP-posteriors	20.7	17.1	15.7	28.4

Table 3.7 Comparison of tandem feature combination and system combination after speaker adaptation using SAT/CMLLR on Quaero Spanish. The total word error rate is divided into the substitution (sub), deletion (del) and insertion (ins) errors. Feature concatenation after linear discriminant analysis reduction is marked by $+$ whereas the system combination result is marked by $\oplus$. The parameters are tuned on the dev10 corpus.

Feature type	Testing corpora (WER [%])							
	dev10				eval10			
	Sub	Del	Ins	Total	Sub	Del	Ins	Total
MFCC	13.3	6.4	2.4	22.0	11.2	5.1	2.0	18.3
MLP-posteriors	13.3	6.8	2.4	22.4	11.2	5.3	1.9	18.5
MFCC + MLP-posteriors	12.2	5.7	2.6	20.4	10.2	4.5	2.1	16.9
MFCC $\oplus$ MLP-posteriors	11.5	7.5	1.8	20.7	9.7	6.1	1.4	17.1

Table 3.7 presents the corresponding detailed word error rates including substitution, deletion and insertion errors for the development and evaluation corpora of 2010. The two combination methods reduce the individual substitution, insertion and deletion errors on almost all corpora. Nevertheless, the systems trained behave differently on the detailed word error rates. System combination tends to produce more deletion and fewer substitution errors, whereas the feature combination approach of the MFCC and the multi-layer perceptron based features results in more insertion errors and a higher substitution error rate compared to the system combination result.

3.2.1.2 Feature Transforms

In addition to the baseline system, seven systems are trained grouped in three main blocks. In the first block, the log posterior estimates are augmented to the linear discriminant analysis transformed MFCC features without any further transformation or reduction. The feature extraction of the second block follows the procedure shown in Figure 3.2 (c). The MFCCs and the multi-layer perceptron posterior features are first augmented and projected to a 68 dimensional feature space by linear discriminant analysis afterwards. Whereas the context information of 9 consecutive frames is taken into account for the MFCCs, a symmetric window of size 1 or 9 frames is applied to the multi-layer perceptron log posteriors. In the last block, A principal component analysis/linear discriminant analysis transforms the MFCC features as well as the posterior features independent of each other. The concept is illustrated in Figure 3.2 (b). Table 3.8 summarizes the recognition results of the three main feature combination methods.

Independent of the feature extraction method used and without any contextual information for the posterior features, all systems achieve a similar performance after speaker adaptation (see Table 3.8). Table 3.9 shows a completly different behavior for the non-speaker adapted systems. There, decorrelation of the input features helps to obtain the best performance. The matrices estimated for speaker adaptation cope with the missing decorrelation steps and apply the missing transformation to the features. By providing temporal context during the decorrelation step of the input features, a small gain in performance could be obtained after speaker adaptation.

Table 3.8 Comparison of several feature combinations using a single matrix or transformation matrices for each feature stream. The tandem recognition results are speaker adapted using SAT/CMLLR. The multi-layer perceptron-posteriors are trained on the MFCCs and the 33 phoneme classes as targets. The multi-layer perceptron-posteriors are augmented by the MFCCs. The MFCCs are reduced by linear discriminant analysis to 45 components including a temporal context of ± 4 frames. The multi-layer perceptron based posteriors are transformed by principal component analysis or linear discriminant analysis using a temporal context of ± 0 or ± 4 frames. Furthermore, a single linear discriminant analysis matrix is used to combine both streams.

	GHMM input feature transformation				Testing corpora (WER [%])			
	MFCC	MLP-posteriors	Context	Final Size	dev10*	eval10	eval09	dev09
MFCC	LDA	—	—	45	21.6	18.2	16.7	29.8
+ MLP	LDA	—	± 0	78	20.7	17.2	15.7	29.0
		PCA	± 0	68	20.7	17.0	15.6	28.6
		LDA			20.4	16.9	15.6	28.6
			± 4	90	20.4	16.9	15.5	28.4
				105	20.4	17.0	15.7	28.7
	Global LDA matrix		± 0	68	20.7	17.1	15.7	28.9
			± 4	90	21.4	17.6	16.4	29.3

Table 3.9 Comparison of several feature combinations using a single matrix or transformation matrices for each feature stream. The multi-layer perceptron-posteriors are trained on the MFCCs and the 33 phoneme classes as targets. The multi-layer perceptron-posteriors are augmented by the MFCCs. The MFCCs are reduced by linear discriminant analysis to 45 components including a temporal context of ± 4 frames. The multi-layer perceptron based posteriors are transformed by principal component analysis or linear discriminant analysis using a temporal context of ± 0 or ± 4 frames. Furthermore, a single linear discriminant analysis matrix is used to combine both streams.

	GHMM input feature transformation				Testing corpora (WER [%])			
	MFCC	MLP-posteriors	Context	Final Size	dev10*	eval10	eval09	dev09
MFCC	LDA	—	—	45	23.3	19.0	17.9	32.2
+ MLP	LDA	—	± 0	78	22.9	18.8	17.3	32.1
		PCA	± 0	68	22.2	18.2	17.1	30.8
		LDA			21.9	18.0	16.8	30.8
			± 4	90	21.9	18.1	16.9	30.7
				105	22.4	18.2	16.9	31.3
	Global LDA matrix		± 0	68	22.2	18.2	17.1	31.3
			± 4	90	22.5	18.5	17.3	31.2

Overall, the best feature combination performance is achieved when linear discriminant analysis transformation matrices for each feature streams are estimated. The size of the final feature vector plays a minor role.

3.2.2 Summary

This section presented an experimental comparison of different feature combination methods. The following conclusions were drawn from the result section:

- ANN based posterior estimates performed as well as short-term features (e.g. MFCCs).
- multi-layer perceptron-posteriors provide information that was contrary to the information of the recognition system.
- Best recognition performance was obtained when MFCC and multi-layer perceptron posteriors were combined.
- Optimal combination: transformation matrices for each feature stream.
- System combination did not lead to the same performance (slightly worse).

Moreover, the different behaviors on the speaker independent and speaker adapted system suggested that it is essential to always compare the system after speaker adaptation. The speaker independent systems will give a hint, but no meaningful conclusion can be drawn from its results.

In order to obtain the optimal performance for all further experiments, the feature combination method as illustrated in Figure 3.2 (b) will be used. System combination results and single feature stream results will be skipped as well as speaker independent recognition results.

3.3 Discriminative Training and Neural Network Features

All conventional training steps of an automatic speech recognition system can be applied to a tandem based system. In addition to speaker adaptive training, discriminative training improves the acoustic model [Heigold 10]. In the discriminative training the discriminative training criterion re-estimates the generative Gaussian hidden Markov models. The maximum mutual information or the minimum phoneme error objective functions are two typical training criteria for string recognition tasks. These methods are based on a (regularized) loss function [Povey & Woodland 02, Heigold 10]. In contrast, large margin classifiers maximize a separate margin resulting in margin-based maximum mutual information or margin-based minimum phoneme error. An additional loss term penalizes the misclassified samples.

As the training of the ANN itself is a discriminative training criterion, the effect of the conventional discriminative training is not clear. On the one hand the improvements of the ANN based systems can be achieved by the discriminative training criterion during the training of the ANNs. On the other hand, the improvements are the result of a better modeling and discrimination of the target classes chosen. In the following, we analyze the effect of the ANN features in the discriminative training step on two languages. We have already shown in [Hoffmeister & Plahl$^+$ 07] that the discriminative training approach improves the ANN based tandem systems.

In this thesis, we apply the margin-based minimum phoneme error criterion to further improve the speaker adapted models. This margin term can be interpreted as an additional observation dependent prior, which weakens the true prior, and is identical with the support vector

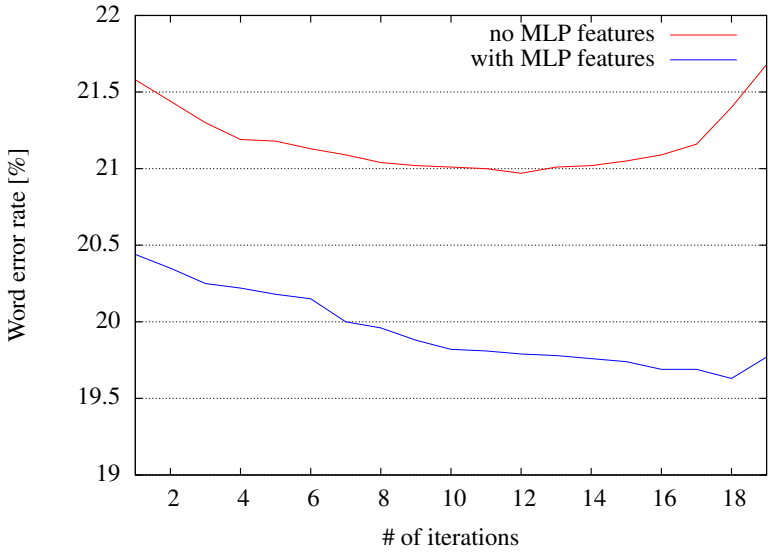


Figure 3.3 Progress of the discriminative training of the speaker adapted acoustic model on the Spanish development set. The acoustic model is trained with and without margin-based minimum phoneme error based features. The optimal configuration is reached after 12 and 18 iterations.

machine optimization problem of log-linear models [Heigold & Deselaers+ 08, Heigold 10]. The modification of the minimum phoneme error training criterion described in [Heigold & Deselaers+ 08] is similar to the margin based extension of the maximum mutual information criterion by [Povey & Kanevsky+ 08].

3.3.1 Experimental Results

In order to analyze the effect of the discriminative training to the multi-layer perceptron based systems, we apply the margin-based minimum phoneme error criterion [Heigold & Deselaers+ 08] to the speaker adapted models of the systems with and without multi-layer perceptron features. We analyze the effect of the discriminative training on two different speech recognition tasks. The multi-layer perceptrons are trained on the Chinese and the Spanish tasks using short-term MFCC features as input. A linear discriminant analysis transforms the final multi-layer perceptron based posterior estimates to a 45 dimensional subspace. Afterwards, the transformed features are combined with the linear discriminant analysis transformed MFCC feature vector. This concept is proven to achieve the best recognition performance, as shown in the previous section.

The number of iterations of the margin-based minimum phoneme error training is optimized according to the recognition performance on the development set of each language. Figure 3.3 illustrates the dependency on the number of margin-based minimum phoneme error iterations and the corresponding recognition performance on the Spanish task. As expected, the margin-based minimum phoneme error training decreases the word error rate on the development set independently of the features used. It is noticeable that the multi-layer perceptron based margin-

Table 3.10 Comparison of different model adaptation methods using SAT/CMLLR based speaker adaptation and margin-based minimum phoneme error (margin-based MPE) discriminative training on Quaero Spanish. The acoustic models are trained with and without multi-layer perceptron-posteriors which use MFCCs as input and the 33 phoneme classes as target classes. The parameters are tuned on the development corpus marked by *.

	GHMM	Testing corpora (WER [%])			
	Input size	dev10*	eval10	eval09	dev09
MFCC	45	23.3	19.0	17.9	32.2
+ SAT/CMLLR		21.6	18.2	16.7	29.8
+ margin-based MPE		21.0	17.4	16.4	28.9
+ MLP-posteriors	90	21.9	18.0	16.8	30.8
+ SAT/CMLLR		20.4	16.9	15.5	28.4
+ margin-based MPE		19.6	16.5	15.1	27.8

Table 3.11 Comparison of different model adaptation methods using SAT/CMLLR based speaker adaptation and margin-based minimum phoneme error (margin-based MPE) discriminative training. The acoustic models are trained with and without multi-layer perceptron-posteriors on Chinese. The multi-layer perceptron-posteriors use MFCCs as input and the 33 phoneme classes as output. The parameters are tuned on the development corpus marked by *.

	GHMM	Testing corpora (CER [%])			
	Input size	dev07*	dev08	eval08	eval07-seq
MFCC	45	15.7	14.6	19.6	16.5
+ SAT/CMLLR		13.8	12.9	17.4	14.7
+ margin-based MPE		12.8	12.3	16.3	14.0
+ MLP-posteriors	90	14.5	13.9	18.3	15.6
+ SAT/CMLLR		12.7	12.4	16.3	14.0
+ margin-based MPE		12.2	11.8	15.5	13.6

based minimum phoneme error training needs more iterations to reach the optimal configuration than the system without the multi-layer perceptron based features. Nevertheless, in the Chinese system the number of iterations is the same for both systems. Independently of the features used, the discriminative training approach improves each system.

Table 3.10 and Table 3.11 summarize the final best recognition results of the margin-based minimum phoneme error trained acoustic model on Spanish and Chinese task respectively. Whereas the recognition performances on Spanish are given by the word error rate, the performances for Chinese are presented using the character error rate.

The experimental results show that all systems are improved by the discriminative training independently of the feature used to train the acoustic model. The improvements become smaller on the testing corpora than on the development corpus which has been used to evaluate the margin-based minimum phoneme error training iterations. Depending on the corpus used, the acoustic model improves up to 4% relative in word error rate on the Spanish task and up to 7% relative in character error rate on the Chinese task. Nevertheless, the systems with and without

the multi-layer perceptron based features obtain the same relative improvements. It is notable that the discriminative trained baseline system —no neural network based features are used— does not outperform the speaker adapted result of the systems using multi-layer perceptron based features. Even more, the speaker adapted system using multi-layer perceptron based features achieves better recognition results on Spanish than the margin-based minimum phoneme error trained baseline system.

On the basis of these results, it can be said that it is not only the discriminative training criterion during the ANN training that caused the achievements of the multi-layer perceptron based features in the tandem approach. They are obtained mostly from the better way to discriminate the phoneme classes and to provide this information as input to the Gaussian hidden Markov model.

3.3.2 Summary

In this section we analyzed the effect of the margin-based minimum phoneme error based discriminative training criterion on tandem based systems. Independently of whether multi-layer perceptron features were used or not, the discriminative training improved the acoustic model and the overall performance.

Even though the difference of the systems with and without multi-layer perceptron features became smaller, a large gap between the systems remained. The relative improvements of the multi-layer perceptron based system by the discriminative training approach were larger on the development set than on all other sets. Overall, the improvements by margin-based minimum phoneme error were significant on all corpora and languages and systems.

Based on the obtained similar effect of the discriminative training of systems with and without multi-layer perceptron based features, a fair comparison of the system was possible even though discriminative training was missing. Therefore, we will skip the discriminative training in all further experiments and compare the speaker adapted systems only.

In order to obtain the best performance of an automatic speech recognition system, discriminative training was necessary even when features were provided which had already been trained in a discriminative manner like the ANN based probabilistic features.

3.4 Relevance of Input Features for Neural Network Training

The tandem approach is a very easy and efficient concept to integrate probabilistic features derived from an ANN into state-of-the-art automatic speech recognition systems. In speech recognition as well as in image recognition and other areas several feature pre-processing steps are known. The pre-processing steps of the features eliminate irrelevant data and select the important information using additional knowledge. This leads to better classification results. Using these adapted features, better classification results could be obtained.

In this section, we analyze the effect of different feature pre-processing steps for the quality of probabilistic feature derived by an ANN. The analysis is split up into three major parts. In the first part, we investigate the significance of different short-term feature types. In this context MFCC, PLP and GT feature extraction methods are applied to the speech signal. The second

part studies the influence of different adaptation techniques, e.g. vocal tract length normalization or constrained maximum likelihood linear regression to the features. In the last part, the multi-layer perceptron posteriors are trained on long temporal context of 500ms or more and their relevance are tested. Experimental results are obtained on the Spanish data set and verified on Chinese.

The section is structured as follows: We start by analyzing different short-term features for the training of the multi-layer perceptron in Section 3.4.1. Several different feature adaptation techniques like vocal tract length normalization and constrained maximum likelihood linear regression are tested in Section 3.4.2. In Section 3.4.3 the significance of the temporal pattern based features as well as the multi-resolution RASTA features are investigated. Finally, we conclude the obtained results and verify them on Chinese in Section 3.4.4

3.4.1 Short-term Features

The classical features used for automatic speech recognition tasks are the MFCC and PLP features. The MFCC and PLP features are first introduced in [Davis & Mermelstein 80] and in [Hermansky 90]. [Zolnay 06] gives detailed information about the different feature sets.

Next to MFCCs and PLPs, systems based on GT features are trained. The GT feature extraction used here is first publicized in [Schlüter & Bezrukov$^+$ 07]. Auditory filter banks realized by Gammatone filters extract these features. The filters are defined in the time domain instead of the frequency domain as for MFCCs or PLPs. In [Plahl & Hoffmeister$^+$ 09] we have successfully introduced the concept of vocal tract length normalization for these GT features. Nevertheless, vocal tract length normalization transformed features are not used in this thesis.

3.4.1.1 Experimental results

A system for each of the short-term feature sets used is set up as a baseline. Table 3.12 lists the recognition performance results of these baseline systems on the Spanish Quaero task. Each of the systems trained performs competitive to each other. Therefore, we expect that multi-layer perceptron posterior estimates trained on the individual feature sets achieve similar performance, too. Detailed information on the acoustic training and the multi-layer perceptron configurations are given in Section A.3

The tandem systems based only on the multi-layer perceptron posterior estimates, which are trained on different short-term features, achieve similar performances. Compared to the results of the baseline systems, the tandem systems are only slightly worse. Table 3.13 summarizes the tandem recognition results. It is notable that the neural network based posteriors trained of GT features achieve the best word error rate on the development set. As shown in Table 3.13, the generalization of other multi-layer perceptron based features to unknown data is much better.

Figure 3.4 shows the progress of the frame accuracy rate on the training set as well as the validation set during the multi-layer perceptron training. The training and the validation set contain 53M and 4M frames respectively. As shown, slight differences in the frame accuracies are obtained after the multi-layer perceptron training. This makes predicting the behavior in the final recognition system hard and only allows to approximately assuming how well the

Table 3.12 Comparison of different input features for Gaussian hidden Markov model based systems on Quaero Spanish. All input features are transformed by linear discriminant analysis, including a temporal context of ± 4 frames, and speaker adaptation using SAT/CMLLR.

	Testing corpora (WER [%])			
Feature type	dev10	eval10	eval09	dev09
MFCC	22.0	18.3	16.8	30.4
+ voiced	22.0	18.4	16.9	30.1
+ VTLN	21.6	18.2	16.7	29.8
+ voiced	21.7	18.1	16.6	29.6
PLP	22.2	18.5	17.0	29.5
GT	21.7	18.2	16.6	29.2

Table 3.13 Comparison of different short-term features used for multi-layer perceptron training on Quaero Spanish. Each tandem system is speaker adapted using SAT/CMLLR. A linear discriminant analysis reduces the multi-layer perceptron-posteriors to 45 components, including a temporal context of ± 4 frames.

MLP	Testing corpora (WER [%])			
Input	dev10	eval10	eval09	dev09
MFCC	22.4	18.5	17.1	30.7
PLP	22.6	18.8	16.9	30.1
GT	22.0	18.8	17.2	29.7

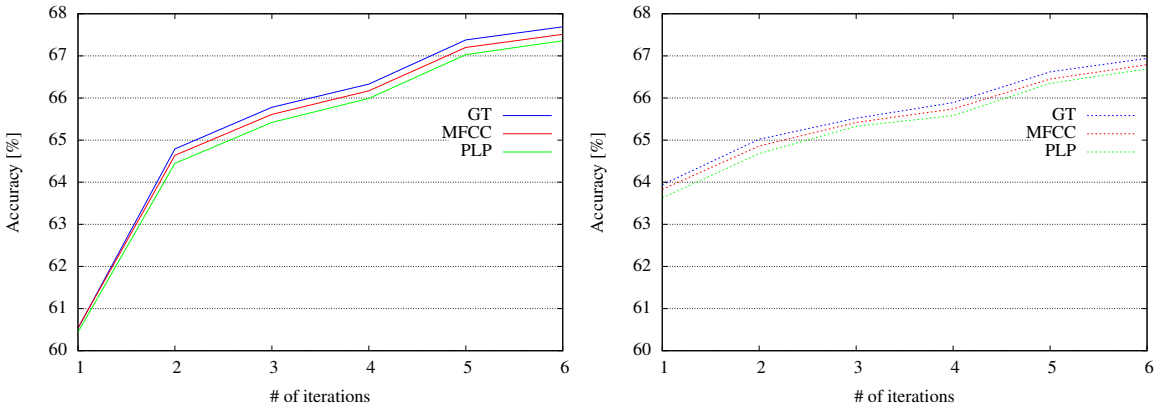


Figure 3.4 Progress of the frame accuracies during the multi-layer perceptron training on the training (left) and validation set (right). The multi-layer perceptrons are trained on different short-term features on the Quaero Spanish task and use the 33 phonemes as target classes. The short-term features cover MFCCs, PLPs and GTs.

Table 3.14 Comparison of different short-term features for Gaussian hidden Markov model and multi-layer perceptron training. The tandem systems are trained on the short-term features and multi-layer perceptron-posteriors which are based on the short-term features as well. In the tandem system the multi-layer perceptron-posteriors and the short-term features are reduced by linear discriminant analysis to 45 components, including a temporal context of ± 4 frames. The results are obtained after speaker adaptation using SAT/CMLLR.

Input Feature	MLP input Type	GHMM Input size	Testing corpora (WER [%]) dev10	eval10	eval09	dev09
MFCC	—	45	22.0	18.3	16.8	30.4
PLP	—		22.2	18.5	17.0	29.5
GT	—		21.7	18.2	16.6	29.2
MLP-posteriors	MFCC	45	22.4	18.5	17.1	30.7
	PLP		22.6	18.8	16.9	30.1
	GT		22.0	18.8	17.2	29.7
MFCC + MLP-posterior						
	MFCC	90	20.4	16.9	15.5	28.4
	PLP		20.6	16.9	15.4	28.1
	GT		20.4	16.8	15.7	27.8

posteriors will work. The multi-layer perceptron training is performed using a hidden layer of size 4000.

Table 3.14 summarizes the baseline results as well as the multi-layer perceptron based posterior results. In addition, we have combined the two streams and trained a tandem system on top. The MFCC feature stream and the posterior estimates are augmented as illustrated in Figure 3.2 (b). Therefore, a LDA transforms each feature stream before they are concatenated afterwards. The linear discriminant analysis transformation includes a temporal context of ± 4 frames. As mentioned before, the tandem multi-layer perceptron posterior systems are only slightly worse. Even if the difference is about 0.4% absolute worse in word error rate on the development set, the difference is significantly less on the evaluation corpora.

The combined feature stream systems benefit from the multi-layer perceptron posteriors as well as from the short-term MFCC features. The baseline systems are improved by 6%-8% relative resulting in $1.3 - 1.6\%$ absolute in word error rate. After speaker adaptation these systems perform competitive to each other on all testing corpora. Moreover, the differences of the individual feature streams do not play any significant role anymore. Even providing the same short-term features for training the multi-layer perceptron and for their combination afterwards do not result in worse performance (here: MFCCs). This leads to the conclusion that the posterior estimates provide contrary information to the automatic speech recognition system.

3.4.1.2 Summary

In this section we analyzed the influence of several short-term features. During the training of the multi-layer perceptrons we obtained no difference in the frame accuracies on the training and validation set. Nevertheless, the differences in performance of the feature dependent baseline systems were obtained as well on the multi-layer perceptron posterior based systems. Although the variation in the frame accuracies was small, the difference in the recognition performance was significant.

The best performance was achieved when the multi-layer perceptron posteriors and the baseline features were combined to train a tandem system. Whereas in the speaker independent case the performance improvements were visible, after speaker adaptation all different short-term features achieved similar recognition results. The reader should keep in mind that presenting the same feature set for training the neural network and the final tandem system was as successful as using another feature set for training the network.

Using this background knowledge, the choice of the single short-term feature used became insignificant. Nevertheless, different ANN trainings could be compared directly taking the frame accuracy into account. A complete training was necessary in order to select the best tandem system and to make a fair comparison.

3.4.2 Feature Adaptation

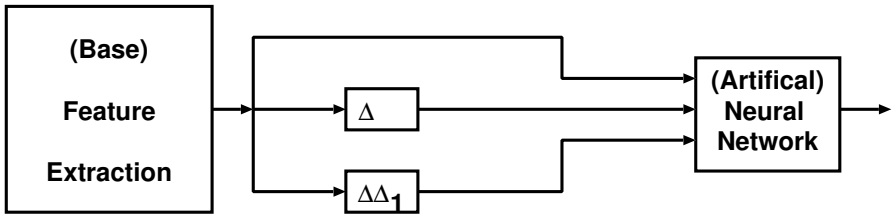
In the previous section different feature sets have been investigated. In this section we analyze the influence of different post processing steps of these features. The short-term features used to train the multi-layer perceptron do not result in any difference after speaker adaptation. Therefore, we focus on the MFCC feature stream and apply the following post processing steps:

- Linear discriminant analysis
- Vocal tract length normalization
- Speaker adaptive training

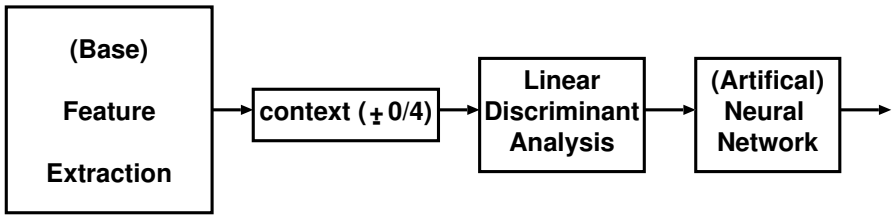
The section is structured as follows: First, we briefly introduce the three post-processing and adaptation methods and in Section 3.4.2.4 we evaluate the corresponding experimental results.

3.4.2.1 Linear Discriminant Analysis

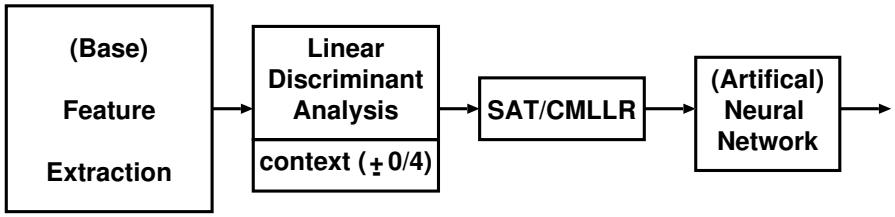
The linear discriminant analysis is a linear transformation which finds a projection to a lower dimensional sub space maximizing the class separability of the underlying distribution [Duda & Hart$^+$ 01, Fink 03]. In speech recognition the linear discriminant analysis is applied to several consecutive feature vectors instead of using just a single feature vector. Therefore, the linear discriminant analysis estimates an optimal linear combination of successive feature vectors [Haeb-Umbach & Ney 92]. The implementation details of the linear discriminant analysis used are described in detail in [Zolnay 06] and additional information is given in [Duda & Hart$^+$



(a) Normal feature extraction with Δ and $\Delta\Delta_1$



(b) linear discriminant analysis transformed features



(c) Feature extraction including speaker adaptation

Figure 3.5 Modification of the feature extraction used for multi-layer perceptron training. In the first step, the first derivative (Δ) and the second derivatives of the first component ($\Delta\Delta_1$) are replaced by the linear discriminant analysis transformed MFCCs (b). In the second step, speaker adaptation using constrained maximum likelihood linear regression is applied to the linear discriminant analysis transformed features (c).

01, Fink 03]. As shown in [Beulen & Welling$^+$ 95, Zolnay 06], the automatic speech recognition system is improved by a linear discriminant analysis transformation instead of using the first and second derivatives.

In order to benefit from the linear discriminant analysis transformation, the preprocessing of the input features of the linear discriminant analysis training is changed. As shown in Figure 3.5 the estimation of the first derivatives (Δ) and the first component of the second derivatives ($\Delta\Delta_1$) are replaced by the linear discriminant analysis transformation. The linear discriminant analysis is calculated on 9 consecutive frames and reduces the feature dimension from $16 * 9 = 144$ down to 45 components. Finally, the multi-layer perceptron training takes nine consecutive linear discriminant analysis transformed frames as input resulting in a final input size of 405 feature components. No additional changes in the multi-layer perceptron setup are necessary. The hidden layer is fixed to 4000 nodes and the training is performed on the 33 phoneme targets of the Spanish corpus.

3.4.2.2 Vocal Tract Length Normalization

In a vocal tract length normalization system, the frequency axis is warped during the calculation of the MFCC coefficients. Since a huge part of the variability in the speech signal is caused by the speaker dependent vocal tract length, vocal tract length normalization tries to normalize this effect by warping the frequency axis of the power spectrum of a speech segment. Several warping functions have been proposed to model the speaker dependent changes of the frequency axis. In [Wegmann & McAllaster+ 96] a piecewise linear function is tested and in [Acero 90] a bilinear transformation. In the RASR system [Rybach & Gollan+ 09] the piecewise linear function works best [Zolnay 06]. There, a speaker dependent warping is carried out first to account for speaker dependent vocal tract length. After the vocal tract length normalization warping, the standard Mel warping is applied carrying out the second warping step. Nowadays, a lot of new aspects of vocal tract length normalization are analyzed and optimized [Sanand & Schlüter+ 10].

As shown in many publications [Zolnay 06, Sanand & Schlüter+ 10] vocal tract length normalization improves the system performance of a speech recognition system (see Table 3.6 on page 35). In this thesis we apply a fast one-pass variant of vocal tract length normalization to the filter bank within the MFCC extraction both in training and testing. The fast vocal tract length normalization performs warping factor estimation using Gaussian hidden Markov models trained on a subset of the training corpus, for which warping factors are estimated using a usual grid search.

In order to benefit from these improvements, the multi-layer perceptron training is performed on the vocal tract length normalization warped MFCC features. Since the feature size is not affected by the vocal tract length normalization transformation, the same multi-layer perceptron training setup is used. The vocal tract length normalization warped MFCC features are calculated beforehand and normalized by mean and variance. Afterwards, the features are augmented by the first and second derivatives, thus including temporal changes.

3.4.2.3 Speaker Adaptation

Speaker adaptation is applied as the last preprocessing step. We transform the input features using constrained maximum likelihood linear regression [Gales 98]. All state-of-the-art automatic speech recognition systems include speaker adaptation to compensate for the acoustical variations due to speaker differences. Instead of using the standard approach, we apply the simple target model approach [Stemmer & Brugnara+ 05]. As target model an acoustic model with a single Gaussian per state are trained on warped and non-warped MFCC features.

To provide a speaker labeling, we apply a generalized likelihood ratio based segment clustering with a Bayesian information criterion based stopping condition [Chen & Gopalakrishnan 98]. The segmented and clustered corpus is used afterwards to estimate the constrained maximum likelihood linear regression matrices needed by the adaptation step in training and decoding.

As shown in Figure 3.5 (c), we modify the linear discriminant analysis based feature extraction to include the speaker adapted features in the multi-layer perceptron training. The whole

Table 3.15 Comparison of different feature adaptations techniques for multi-layer perceptron training on Quaero Spanish. The adaptations include vocal tract length normalization (VTLN), linear discriminant analysis (LDA) and speaker adaptation using constrained maximum likelihood linear regression (CMLLR). The tandem system is trained on the MFCC transformed by linear discriminant analysis, including a temporal context of ± 4 frames augmented by the multi-layer perceptron-posteriors. The multi-layer perceptron-posteriors are reduced by principal component analysis to 23 components.

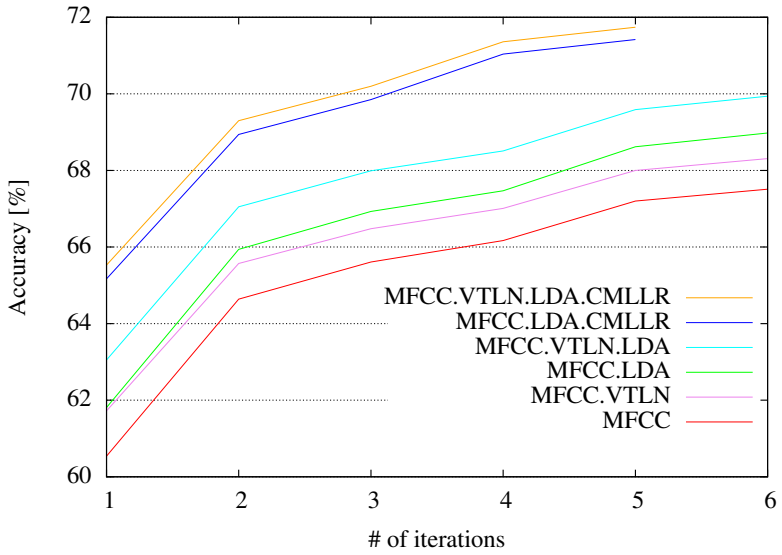
	MLP input feature		Testing corpora (WER [%])			
	Type	Size	dev10	eval10	eval09	dev09
MFCC	—	—	23.3	19.0	17.9	32.2
+ MLP-posteriors	MFCC	33	22.4	18.2	16.9	31.3
	+ LDA	45	22.1	18.2	16.9	31.3
	+ CMLLR		21.2	17.8	16.3	30.1
	+ VTLN	33	22.0	18.3	16.9	30.7
	+ LDA	45	21.9	18.1	16.8	30.5
	+ CMLLR		21.3	17.6	16.3	29.8

setup of the multi-layer perceptron keeps unchanged. The hidden layer contains 4000 nodes and in the output layer the softmax activation function is applied to the 33 phonetic targets of the Spanish system.

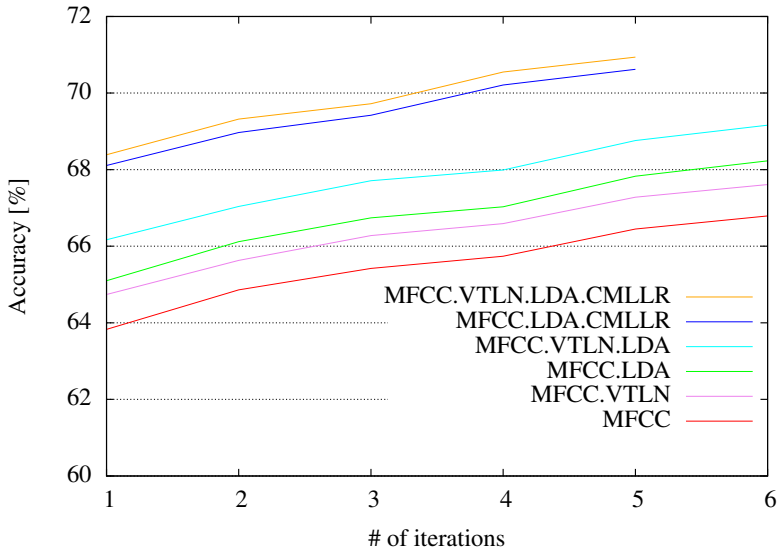
3.4.2.4 Experimental Results

As shown in Figure 3.6, the adapted features, used as input to train the multi-layer perceptron, improve the frame accuracies on the training and validation set. The multi-layer perceptron is trained on the Spanish Quaero training corpus, which is described in detail in Section A.3. On the one hand, the best results during the multi-layer perceptron training are obtained, when different adaptations are stacked. One the other hand, the observed gain on the vocal tract length normalization warped MFCC features is small. Nevertheless, the linear discriminant analysis transformed features are improved by more than 1.5% absolute in word error rate on the training set and the best frame accuracy is achieved when the multi-layer perceptron is trained on speaker adapted features. The baseline multi-layer perceptron is improved from a frame accuracy of 67.5 to 71.4 and 66.8 to 70.6 on the training and validation set respectively. When the multi-layer perceptron is trained on all three adaptations, the frame accuracy rises from 68.3 to 71.7 and 67.6 to 70.9 on the training and validation set.

The same observation can be made when a speaker independent model is trained. The multi-layer perceptron posteriors trained on the speaker adapted input achieve the best tandem recognition results. Even though a noticeable performance gap in the multi-layer perceptron training exists, both speakers adapted tandem systems behave similarly. Only on the evaluation set of 2010 the non-warped speaker adapted systems result in slightly worse results. All speaker independent tandem systems are trained on linear discriminant analysis transformed MFCCs augmented with principal component analysis transformed multi-layer perceptron posteriors (see Figure 3.2 (b)). As mentioned in Section 3.2, the transformation used to reduce the feature dimension of the multi-layer perceptron posteriors plays a minor role. The principal component analysis is trained on one single frame without any context information and reduces the



(a) Training set



(b) Validation set

Figure 3.6 Frame accuracies during the training of the multi-layer perceptron. The accuracies are measured of a training and validation set. Different adaptation techniques preprocess the MFCC features. The feature adaptation steps include vocal tract length normalization (VTLN), linear discriminant analysis (LDA) and speaker adaptation using constrained maximum likelihood linear regression (CMLLR).

Table 3.16 Comparison of different feature adaptation techniques for multi-layer perceptron training on Quaero Spanish after speaker adaptation using SAT/CMLLR. The feature adaptations include vocal tract length normalization (VTLN), linear discriminant analysis (LDA) and speaker adaptation using constrained maximum likelihood linear regression (CMLLR). The tandem system is trained on the MFCC transformed by linear discriminant analysis, including a temporal context of ± 4 frames augmented by the multi-layer perceptron-posteriors. The multi-layer perceptron-posteriors are reduced by principal component analysis to 23 components.

	MLP input feature		Testing corpora (WER [%])			
	Type	Size	dev10	eval10	eval09	dev09
MFCC	—	—	21.6	18.2	16.7	29.8
+ MLP-posteriors	MFCC	33	20.7	17.0	15.5	28.6
	+ LDA	45	20.4	17.0	15.5	28.6
	+ CMLLR		20.5	16.9	15.6	28.7
	+ VTLN	33	20.5	16.9	15.5	28.2
	+ LDA	45	20.4	16.7	15.5	28.2
	+ CMLLR		20.4	16.9	15.6	28.5

33 posteriors to 23 components. Overall, a 68 dimensional feature vector is presented for the Gaussian hidden Markov model training. Table 3.15 summarizes the speaker independent recognition results.

The speaker adapted tandem systems are trained using constrained maximum likelihood linear regression. The input of the system is the linear discriminant analysis transformed MFCC features augmented by the principal component analysis transformed log-posteriors. The augmented features are further transformed by speaker adapted matrices, estimated in training and recognition. In contrast to the previous experimental results shown, the multi-layer perceptron posteriors, trained on different adapted features, do not improve the speaker adapted recognition systems. All systems achieve a similar recognition performance within a range of $\pm 0.1\%$ absolute. This is consistent on all testing corpora. Table 3.16 summarizes the experimental recognition results for the speaker adapted models are.

The slight performance differences of the adapted features are not surprising, since all adaptations used can be expressed as a single linear transformation. When a transformation of the features is missing, the speaker adaptation matrices can cope with the missing transformations. Moreover, speaker adaptation of speaker adapted features does not seem to be a useful combination. Therefore, the adapted features could be used to improve the hybrid recognition performance, but in the tandem system the adapted features are not meaningful after speaker adaptation. This can be changed with the help of the bottle-neck concept, but this will not be analyzed here. The bottle-neck provides an abstract representation of the ANN input features as shown in Section 4.3.

3.4.2.5 Summary

This section analyzed the performance of the multi-layer perceptron posteriors trained on different adapted features. All adaptations applied to the baseline MFCC feature set were expressed as a sequence of linear transformations. The adaptations used include vocal tract length normalization, speaker adaptation as well as linear discriminant analysis.

During the training of the multi-layer perceptron, the frame accuracy on the training and validation sets were improved noticeably. Each transformation of the baseline features resulted in an observable performance gain on the training and validation set. We achieved the best accuracies when all three adaptations (vocal tract length normalization, linear discriminant analysis, constrained maximum likelihood linear regression) were applied. In this case the obtained frame accuracies were increased by 4% absolute.

The same performance gains were not observed on the trained tandem systems. Whereas differences were observed in the speaker independent systems, after speaker adaptation all the gains were not recognizable anymore and all systems showed very similar performance on all corpora. Therefore, speaker adaptation using SAT/CMLLR absorbed the differences in the adapted feature sets and equalized the features used for training.

The different performances on the speaker independent models were observed. There, the tandem systems benefitted from the improved ANN feature extraction. Thus, these improved features were used for hybrid recognitions, where currently a single recognition pass was performed only.

Overall, feature adaptation techniques like vocal tract length normalization and speaker adaptation were not useful to improve the tandem system performance. Moreover, a Gaussian hidden Markov model based speaker adapted acoustic model had to be trained to obtain the matrices for speaker adaptation first.

3.4.3 Long Temporal Features

As mentioned above, a huge variability of different feature extraction methods are known. The classical features used for automatic speech recognition tasks are the short-term MFCC features. In recent years, long-term features modeling long time pattern and dependencies of the speech signal are successfully applied to speech recognition [Hermansky & Sharma 98, Hermansky & Fousek 05, Valente & Vepa$^+$ 07]. These features contain temporal context of about 500ms up to 1000ms and are motivated to model long temporal dependencies in the speech signal.

The results reported in this section summarizes some aspects of the joint work published in [Valente & Magimai-Doss$^+$ 11]. The work analyzes a huge number of different input features as well as different multi-layer perceptron topologies.

3.4.3.1 Short term features and long temporal context

Short-term features, e.g. MFCCs [Davis & Mermelstein 80], are the standard features for automatic speech recognition. Most of these features are based on the frequency spectrum of the spoken utterance. A detailed overview of the short-term features are also given in [Zolnay 06].

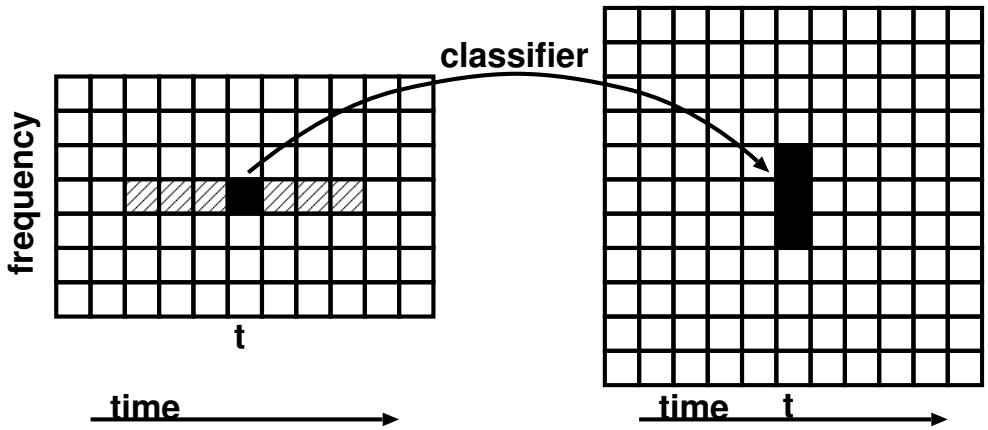


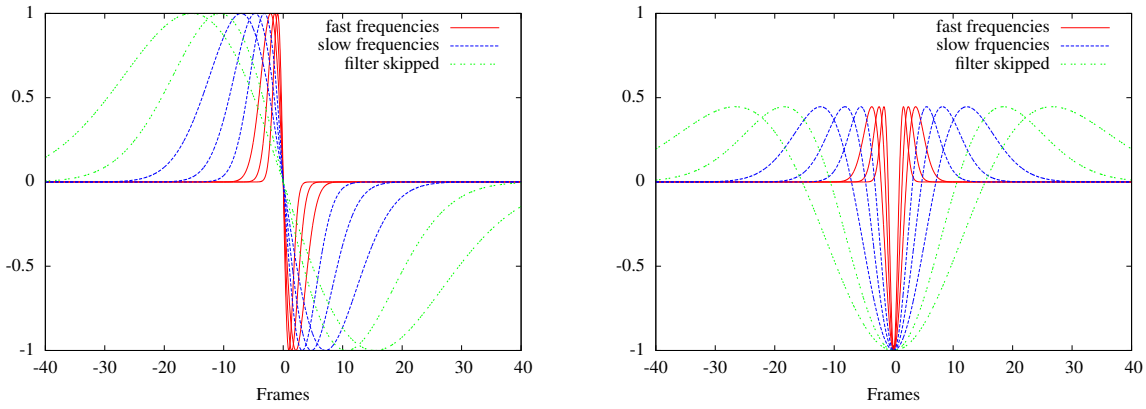
Figure 3.7 Extraction of temporal pattern from the speech signal. For each subband, temporal patterns are extracted independently. The dimension of the the temporal pattern based features is reduced by discrete cosine transform or principal component analysis.

The number of consecutive frames is increased to include large context into the MFCC features. Instead of a maximum of 9 frames, as mentioned in the previous sections, ± 25 frames of the current time step are taken into account which results in an overall temporal context of 500ms encoded in 51 frames. The training of the multi-layer perceptron is performed on a $16 * 51 = 816$ dimensional feature vector. The number of nodes in the hidden layer is fixed to 4000.

3.4.3.2 Classifiers of Temporal Pattern

Instead of presenting short-term MFCC features as input, the multi-layer perceptron is fed by critical band temporal trajectories. These trajectories contain information from up to half a second of the acoustic signal. The aim of these trajectories is to model long-term dependencies of the speech signal, also known as temporal patterns [Hermansky & Sharma 98]. The final feature vector grows very fast with the number of frames used —19 critical band energies and 500ms produces a feature vector of size 950. Many methods are considered for efficiently encoding this information while reducing the dimension [Fousek 07]. This thesis investigates two main concepts, the principal component analysis and a discrete cosine transform resulting in TRAP-PCA and TRAP-DCT, respectively.

The TRAP-DCT and TRAP-PCA features are based on the critical band energies. The critical band auditory spectrum is extracted every 10ms from the short-time fast Fourier transformation of the audio signal. Afterwards, 500ms long energy trajectories for each of the 19 critical bands of the spectrum are constructed. We only use the first 16 coefficients of the discrete cosine transform. Overall, this results in a $19 * 16 = 304$ dimensional feature vector as input for the multi-layer perceptron. In order to compare the discrete cosine transform and the principal component analysis, the principal component analysis reduces the $19 * 51 = 969$ dimensional feature space to 304 components. Figure 3.7 visualizes the temporal pattern approach.



(a) Multi-resolution RASTA filter G_1

(b) Multi-resolution RASTA filter G_2

Figure 3.8 Multi-resolution RASTA filter G_1 and G_2 [Valente & Vepa$^+$ 07]. The fast modulation frequencies are represented by the solid (red) lines and the slow modulation frequencies by the dashed (blue) lines. The two filter response shown in dotted (green) are skipped.

3.4.3.3 Multi-resolution RASTA features

The multi-resolution RASTA filtering [Hermansky & Fousek 05] is an extension of the RASTA filtering [Hermansky 90]. The filters divide the modulation frequency range into its individual subbands. These subbands contain decreasing resolutions moving from fast to slow modulations. In the modulation frequency domain, these filters correspond to a filter-bank with equally spaced filters on a logarithmic scale. The filters are realized by several band pass filters of different modulation frequencies and are represented by first (G_1) and second derivatives (G_2) of Gaussian functions as shown in Equation (3.5) and Equation (3.6) respectively

$$G_{1,i}(x) \approx -\frac{x}{2\sigma_i^2}\ \exp(\frac{-x^2}{2\sigma_i^2}) \tag{3.5}$$

$$G_{2,i}(x) \approx (\frac{x^2}{\sigma_i^4} - \frac{1}{\sigma_i^2})\ \exp(\frac{-x^2}{2\sigma_i^2}) \tag{3.6}$$

with $\sigma_i = \{0.8, 1.2, 1.8, 2.7, 4.0, 6.0\}$ and x as the input.

The variance σ_i of the Gaussian function varies in the range of $8-130$ms. Figure 3.8 show the corresponding filter. Multiple-resolution representations of the time-frequency plane are provided when applied to all critical bands.

Finally, frequency derivatives across three consecutive critical bands are augmented, resulting in additional 204 feature components. In total, $228 + 204 = 432$ feature components are presented as input for the multi-layer perceptron.

Overall, the following steps are applied to obtain the multi-resolution RASTA features:

- Extraction of 19 critical band energies of the auditory spectrum

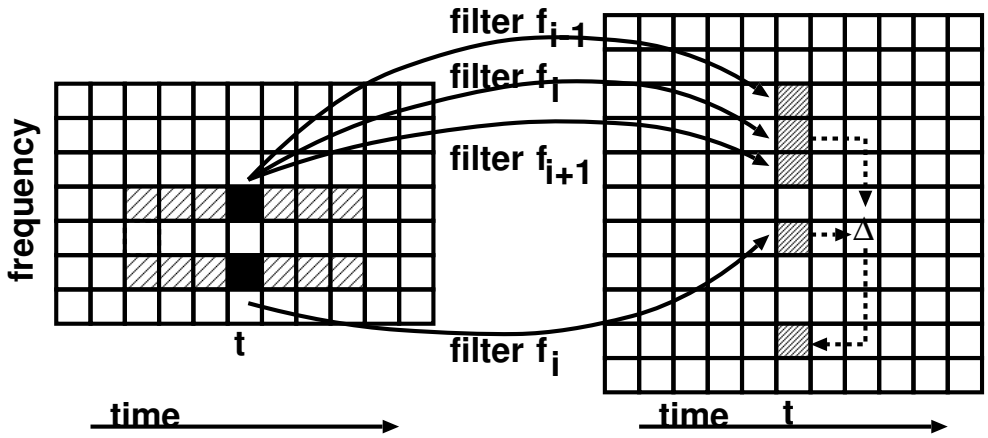


Figure 3.9 Multi-resolution RASTA feature extraction schema. First, we obtain the filter response for each band and filter. Afterwards, the first derivatives of the filter response are calculated. The filter responses as well as the derivatives are combined into a 432 dimensional feature vector.

- Temporal trajectories for each critical band
- Filtering of each trajectory by G_1 and G_2
- Frequency derivatives

Finally, the 432 dimensional feature vector is used for training the multi-layer perceptron as well as for decoding. The different steps are visualized in Figure 3.9.

3.4.3.4 Experimental Results

Several systems are trained using the augmented MFCC feature stream and multi-layer perceptron posteriors trained on the different long-term features mentioned above. As usual, the multi-layer perceptron training is performed using one hidden layer with 4000 nodes and the 33 Spanish phonemes as target classes. Afterwards, the multi-layer perceptron log-posterior estimates are transformed by principal component analysis and are reduced to 23 components. Finally, we train a speaker independent and a speaker adapted system with 1M Gaussians. Section A.3 reports more details of the multi-layer perceptron training and the acoustic training on the Spanish Quaero corpus as well as additional corpus statistics.

Table 3.17 and Table 3.18 summarize the speaker independent and speaker adapted recognition results on the Quaero development and evaluation data. The best recognition performance is achieved by the posteriors trained on the 51 consecutive frames of the MFCC features (MFCC-25) followed by the posteriors trained on the temporal pattern-principal component analysis features. The recognition performances of the other feature sets are up to 0.3% absolute worse. Comparing the input dimension of these two feature types the principal component

Table 3.17 Comparison of different long-term features for multi-layer perceptron training on Quaero Spanish. The multi-layer perceptron based posteriors are reduced by principal component analysis to 23 components and are augmented by the MFCCs reduced by linear discriminant analysis to 45 components, including a temporal context of ± 4 frames.

MLP input feature pre-processing				Testing corpora (WER [%])			
Base type	Context	Filtering	Size	dev10	eval10	eval09	dev09
MFCC	$\pm$ 4	—	297	22.4	18.2	16.9	31.3
	± 25	—	816	22.0	18.1	16.6	30.9
CRBE	± 25	TRAP-DCT	304	22.4	18.3	17.2	30.5
		TRAP-PCA	304	22.1	18.2	16.8	30.1
	± 50	MRASTA	432	22.3	18.2	16.9	30.3

Table 3.18 Comparison of different long-term features for multi-layer perceptron training on Quaero Spanish after speaker adaptation using SAT/CMLLR. The multi-layer perceptron-posteriors are reduced by principal component analysis to 23 components and are augmented by the MFCCs reduced by linear discriminant analysis to 45 components, including a temporal context of ± 4 frames.

MLP input feature pre-processing				Testing corpora (WER [%])			
Base type	Context	Filtering	Size	dev10	eval10	eval09	dev09
MFCC	$\pm$ 4	—	297	20.7	17.0	15.6	28.6
	± 25	—	816	20.3	16.7	15.4	28.4
CRBE	± 25	TRAP-DCT	304	20.9	17.1	15.7	28.2
		TRAP-PCA	304	20.4	16.8	15.5	27.7
	± 50	MRASTA	432	20.6	17.0	15.7	28.2

analysis shows to be an efficient and effective method to reduce the input size without losing much of its performance at the end.

The recognition performances of the other systems trained are competitive to each other. The worst result is obtained by the temporal pattern-discrete cosine transform features, whereas the MFCC-4 and multi-resolution RASTA feature perform similarly on the development data of 2010 and the evaluation data of 2009 and of 2010. Since the input dimension of the multi-resolution RASTA features as well as the feature extraction is computationally more expensive than the MFCC-4 system, the MFCC based system will be preferred. Nevertheless, the multi-resolution RASTA system could be improved by the hierarchical approach described in Section 4.2.

3.4.3.5 Summary

This section investigated the necessity of several long-term features. The aim of these long-term features was to model long-term patterns in the speech signal and to provide this information during training. We analyzed and compared the temporal pattern and multi-resolution RASTA feature preprocessing methods and provided the full window without any preprocessing as input.

Table 3.19 Comparison of different short-term and long-term features for multi-layer perceptron training on the Gale Chinese corpus after speaker adaptation using SAT/CMLLR. The tandem systems are trained on MFCCs augmented by the multi-layer perceptron-posteriors. Each feature set is reduced by linear discriminant analysis to 45 components, including a temporal context of ± 4 frames. The Hier-MRASTA features are derived from a hierarchical processing of two multi-layer perceptrons, presenting the fast and slow modulation frequencies of multi-resolution RASTA at different stages of the multi-layer perceptron training.

	MLP input Feature type	Testing corpora (CER [%])			
		dev07	dev08	eval08	eval07-seq
MFCC	—	13.8	12.9	17.4	14.7
+ MLP-posteriors	MFCC	12.7	12.4	16.3	14.0
	PLP	12.6	12.2	16.2	13.7
	GT	12.1	11.7	15.5	13.5
	TRAP-DCT	13.3	12.8	17.1	14.6
	MRASTA	13.4	13.0	17.1	14.4
	Hier-MRASTA	12.9	12.5	16.1	13.6

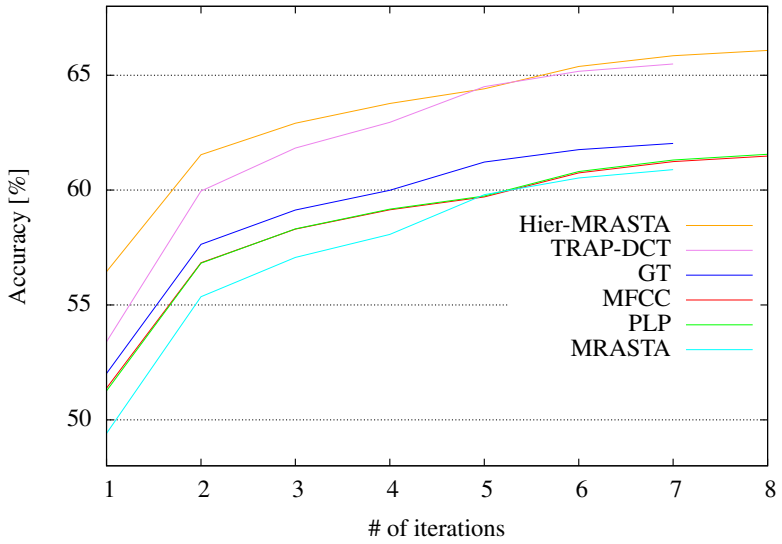
As shown, the discrete cosine transformed long-term temporal pattern and the multi-resolution RASTA features were not superior to the short-term MFCC features based on 9 frames. All the systems achieve similar performance within a range of $\pm$ 0.1% absolute. The results were consistent for the speaker independent and speaker adapted models trained. However, the temporal pattern-principal component analysis based features and the MFCCs with a temporal context of $\pm$ 25 frames obtained the best recognition performances. The MFCCs achieved a consistently better word error rate compared to the principal component analysis transformed temporal pattern features. Therefore, the multi-layer perceptron benefitted from the information skipped after principal component analysis reduction. Compared to the MFCC feature set with a context of $\pm$25 frames, the MFCCs with a context of $\pm$4 achieved a 0.3-0.4% absolute worse word error rate on most corpora.

Overall, the use the MFCC features with a context of $\pm$ 4 frames will be recommend because the input dimension was low, the training and decoding of the multi-layer perceptron was fast and the performance was much better than of any (higher dimensional) feature sets.

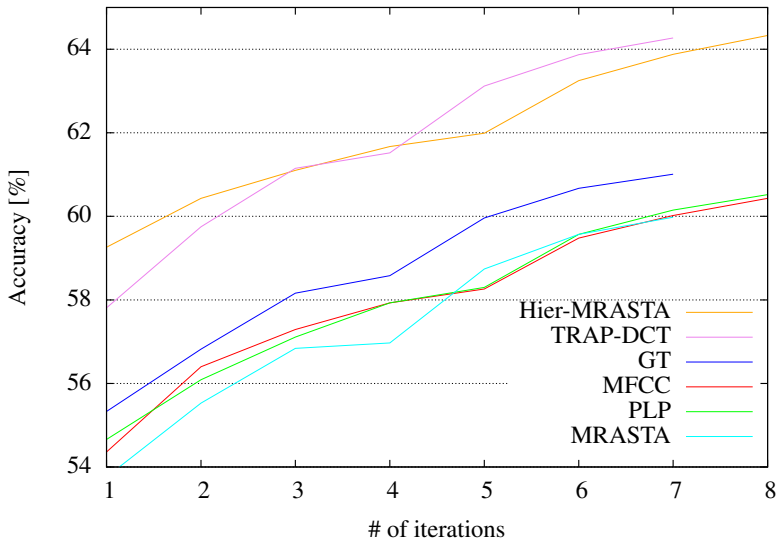
3.4.4 Experimental Verification on Chinese

Table 3.19 shows experimental results on the Chinese cn-small corpus. These experiments verify the results obtained on the Spanish corpus. Instead of using 4k nodes in the hidden layer as for the Spanish task, the number of nodes in the hidden layer is enlarged to 7500. Chapter 11 shows why 7500 nodes should be used in the hidden layers on the Chinese corpus. The number of phonetic targets classes is increased to 71, corresponds to the 71 tonemes used to model the Chinese words. Tonemes are phonemes with tonal information. Section A.1 summarizes the details of the Chinese system used.

As shown, the same behavior of the short-term and the long-term features is observed on the Chinese task. The MFCC baseline system is improved by around 10% relative for all corpora when augmented with the posterior features.



(a) Training set



(b) Validation set

Figure 3.10 Frame accuracies during the multi-layer perceptron training on the Chinese data set. The accuracies are measured of the training and validation set. The multi-layer perceptrons uses short-term and long-term features as input. The hierarchical training of a cascade of two networks with multi-resolution RASTA as input is marked by Hier-MRASTA.

On the Chinese task, the gap between the short-term and long-term features is increased. Whereas on the Spanish task the different feature sets achieve similar performance, the short-term features outperform the long-term features by 0.6% up to 0.8% on all Chinese corpora. Figure 3.10 shows the frame accuracies of the multi-layer perceptron during training. Surprisingly, the frame accuracy of the temporal pattern features is higher, but the corresponding recognition performance is worse.

Considering these insights, the short-term feature is preferred to the long-term features. Moreover, the GT features seem to work better in combination with the MFCC features than any other short-term feature set.

CHAPTER 4

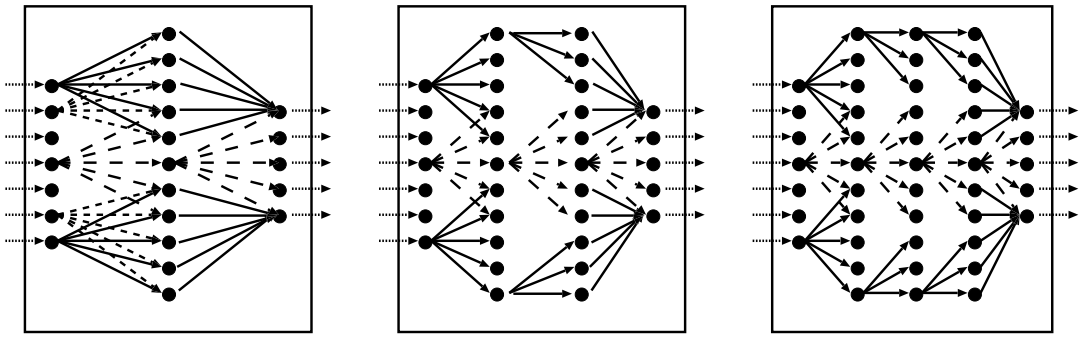
Artificial Neural Network Topologies

In this chapter we introduce and compare different artificial neural network (ANN) processing types as well as different ANN feature extraction methods. Instead of using the posterior estimates derived by an ANN, we can use the node activations of an inner layer of the network. The most promising feature set using this concept are the *bottle-neck* probabilistic features introduced in [Grézl & Karafiat$^+$ 07]. Furthermore, we develop a new processing type by combining the advantages of the bottle-neck processing and the hierarchical ANN concept. The hierarchical framework has been developed in a joint work with Fabio Valente (see [Valente & Vepa$^+$ 07, Valente & Magimai-Doss$^+$ 09]).

The chapter is structured as follows: We start by reflecting on the single ANN processing framework in Section 4.1. In order to improve the posteriors, we expand the ANN training by the hierarchical concept in Section 4.2. We continue by introducing the bottle-neck processing framework in Section 4.3. Finally, in Section 4.4, we describe the new developed ANN processing framework which combines the bottle-neck concept and hierarchical ANNs.

4.1 Single Neural Network Processing

In this section we briefly reflect on the concept of ANN based probabilistic features using a single ANN. Each ANN trained contains an input layer and an output layer. In addition to these layers, each ANN contains one or more hidden layers. The smallest ANN trained in this work uses an input, an output and one hidden layer and the largest ANN consists of three hidden layers. Figure 4.1 illustrates the different single ANN processing types. All the experiments described in Chapter 3 are performed using this single ANN concept with one hidden layer (see Figure 4.1 (a)). As mentioned in the introduction (Chapter 1), a multi-layer perceptron with just one hidden layer can approximate any arbitrary function. Therefore, no additional hidden layers are necessary. Nevertheless, experimental results show that the performance is increased by additional layers.



(a) 2-layer multi-layer perceptron **(b)** 3-layer multi-layer perceptron **(c)** 4-layer multi-layer perceptron

Figure 4.1 Schema of a fully connected feed-forward multi-layer perceptron with one (a) or two (b) or three hidden layers (c).

Table 4.1 Comparison of multi-layer perceptron based features using multiple hidden layers on Quaero Spanish after speaker adaptation using SAT/CMLLR. The tandem systems are trained on MFCC and multi-layer perceptron posterior features. Each stream is reduced by linear discriminant analysis to 45 components, including a temporal context of ± 4 frames. The posterior estimates are derived from a multi-layer perceptron with one, two or three hidden layers trained on MFCCs. During decoding, we tune the parameters on the dev10 corpus.

Feature	Hidden layer		Testing corpora (WER [%])			
Type	#	Size	dev10	eval10	eval09	dev09
MFCC	—	—	21.6	18.2	16.7	29.8
+ MLP-posteriors	1	4000	20.4	16.9	15.5	28.4
	1	1024	20.7	16.9	15.7	28.4
	2		20.0	16.3	15.2	27.5
	3		19.6	16.0	15.0	27.0

The multi-layer perceptrons trained differ in the number of layers and the number of units in the hidden layers. The ANN features are augmented by the baseline MFCC features after each feature stream has been reduced by linear discriminant analysis to 45 dimensions. Table 4.1 summarizes the corresponding tandem recognition results.

In order to simplify the networks, each hidden layer contains the same number of nodes when the number of hidden layers is increased. All networks presented in Table 4.1 are trained on the 33 phonetic target classes of the Spanish task and use MFCCs as input features. As shown, the performance of the multi-layer perceptrons is improved by increasing the number of hidden layers. In this work, we focus on the 2-layer network according to the trade-off performance and computational costs. However, the methods used can be applied to larger networks as well with similar gain in performance.

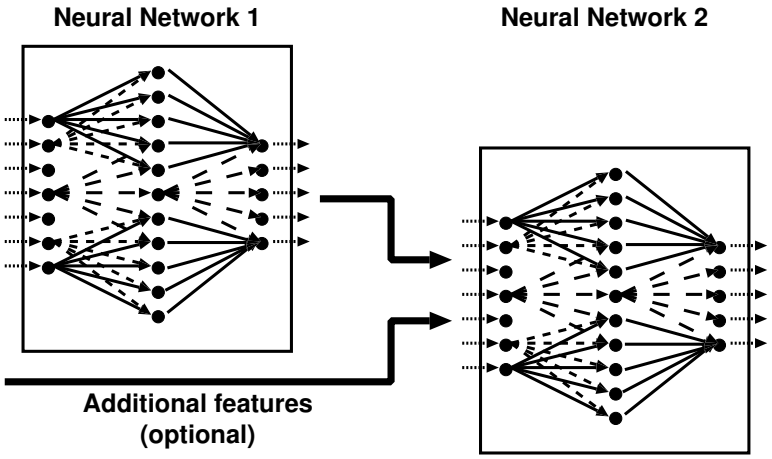


Figure 4.2 Hierarchical ANN processing framework. The output of the first network is used as input for the second network. Optionally, the second ANN can be trained on any other feature set. The target classes of the second network can be changed as well.

4.2 Hierarchical Neural Network Processing

Additional hidden layers lead to significant improvements of the posterior estimates of an ANN. Rather than to increasing the hidden layers, the hierarchical processing concept provides a hierarchy of several networks. These networks are stacked, and the output of a previously trained ANN is used as input for the next ANN training. In the hierarchical processing, a cascade of ANNs is trained, where each ANN in the pipeline uses the output of the previous network. Compared to a single ANN with multiple layers, one big advantage of the hierarchical processing is the possibility to include additional features as well as to increase the temporal context of the features used. An additional advantage is the initialization of large networks. When the number of layers is increased, the training can get stuck in local optima. Chapter 9 discusses the initialization problem in more detail and provides several methods use another initialization of the networks. A third advantage of the hierarchical processing is that the training criterion or the target classes can be changed. In this work, we do not change the target classes and use the same number of target classes and training criterion for all networks.

Figure 4.2 illustrates the general setup of the hierarchical processing framework. There, the hierarchical framework is outlined by two 2-layer feed-forward multi-layer perceptrons. As additional features any feature set can be used. When two different ANN based feature sets are provided, the last multi-layer perceptron works as a merger network. A more efficient method to combine posterior estimates trained on the same classes is to merge the posterior estimates using the Dempster-Shafer theory [Valente & Hermansky 07]. When probabilistic bottle-neck features (see Section 4.3) are derived from the ANN the Dempster-Shafer combination is not suitable.

On a small speech recognition task, the hierarchical concept shows improvements over the baseline [Sivadas & Hermansky 02, Schwarz & Matejka$^+$ 06]. For large vocabulary continuous speech recognition systems this concept has been first published by [Valente & Vepa$^+$ 07]. The performance of an ANN, as well as the overall system performance, is increased by presenting

long-term features to train the first networks and short-term features as well as the posterior estimates from the previous network to train the second network.

We develop this concept further resulting in a hierarchical framework where the fast and slow modulation frequencies of the long-term multi-resolution RASTA features are presented at different training stages [Valente & Magimai-Doss$^+$ 09] in the hierarchy.

4.2.1 Experimental Results

Next to experiments on Arabic [Valente & Vepa$^+$ 07] and Chinese [Valente & Magimai-Doss$^+$ 09], we have performed experiments on the cn-small corpus for Chinese and on the Spanish task. The Chinese and Spanish corpora used are described in detail in Section A.1 and Section A.3 respectively. We apply the hierarchical processing to the multi-resolution RASTA features splitting the multi-resolution RASTA filters into fast and slow modulation frequencies, as described in Section 3.4.3.3.

4.2.1.1 Hierarchical Multi-resolution RASTA Processing

As mentioned above, the multi-resolution RASTA features are divided into fast and slow modulation frequencies, sensible for different time ranges. The multi-resolution RASTA features are derived by different Gaussian filters. Depending on the range of these filters, the fast or the slow frequencies are obtained. Section 3.4.3.3 gives more details on the multi-resolution RASTA feature extraction. In the hierarchical framework, the fast and slow frequencies are provided at different stages. The training of the hierarchical ANNs as well as the final tandem system is performed on the Chinese and Spanish tasks.

In order to focus on the hierarchical framework, the structure of each multi-layer perceptron within the hierarchy will be a simple 2-layer ANN. Each 2-layer ANN for Chinese is trained on the 71 phonetic targets using 7500 nodes in the hidden layer. The first multi-layer perceptron (NN-1) is based on the fast modulation frequencies of the multi-resolution RASTA filtering (F-MRASTA). In the second network (NN-2), the slow modulation frequencies (S-MRASTA) are augmented by the posteriors derived from the NN-1. The NN-1 features are transformed further by logarithm and principal component analysis. Nine consecutive frames are combined and presented as input, thus including temporal context. The frames used are centered at the current time step. The final features obtained from the hierarchical multi-resolution RASTA processing are referred to as *Hier-MRASTA*. Instead of using the multi-resolution RASTA features only, other features like the critical band energies could be augmented in each stage of the hierarchy as well. The extracted features from the hierarchical processing of the multi-resolution RASTA features augmented by the critical band energies are referred to as *A-Hier-MRASTA*. The 19 critical band energies are provided as input for NN-1 as well as for NN-2. Table 4.2 and Table 4.3 summarize the configuration of the Hier-MRASTA and A-Hier-MRASTA processing for Chinese and Spanish.

Table 4.4 and Table 4.5 summarize the experimental results after speaker adaptation on the Chinese and Spanish tasks. As shown, the performance of the speaker adapted system is improved by 0.5% absolute or more in character error rate on all Chinese testing corpora and

Table 4.2 Multi-layer perceptron configuration of the Hier-MRASTA and A-Hier-MRASTA feature extraction for Gale Chinese.

	Hier-MRASTA		A-Hier-MRASTA	
Configuration	NN-1	NN-2	NN-1	NN-2
Input	F-MRASTA	NN-1 S-MRASTA	CRBE F-MRASTA	NN-1 CRBE S-MRASTA
Size	216	423	235	451
Hidden layer	7500	7500	7500	7500
Phonetic targets	71	71	71	71
Posterior transform	LOG+PCA	LOG	LOG+PCA	LOG
Final size	23	71	24	71

Table 4.3 Multi-layer perceptron configuration of the Hier-MRASTA feature extraction for Quaero Spanish.

	Hier-MRASTA	
Configuration	NN-1	NN-2
Input	F-MRASTA	NN-1 S-MRASTA
Size	216	405
Hidden layer	4000	4000
Phonetic targets	33	33
Posterior transform	LOG+PCA	LOG
Final size	23	33

Table 4.4 Comparison of the hierarchical multi-resolution RASTA processing on Chinese after speaker adaptation using SAT/CMLLR. The multi-resolution RASTA features are split into fast and slow modulation frequencies (Hier-MRASTA) and augmented with critical band energies (A-Hier-MRASTA). The final tandem system is trained on MFCCs augmented by the multi-layer perceptron-posteriors. Each feature stream is reduced by linear discriminant analysis to 45 components, including temporal context of size ± 4.

		Testing corpora (CER [%])			
	Feature type	dev07	dev08	eval08	eval07-seq
MFCC	—	13.8	12.9	17.4	14.7
+ MLP-posteriors	MRASTA	13.4	13.0	17.1	14.4
	Hier-MRASTA	12.9	12.5	16.1	13.6
	A-Hier-MRASTA	12.6	12.2	16.0	13.6

Table 4.5 Comparison of the hierarchical multi-resolution RASTA processing on Quaero Spanish after speaker adaptation using SAT/CMLLR. The multi-resolution RASTA features are split into fast and slow modulation frequencies (Hier-MRASTA). The tandem system is trained on MFCCs augmented by the multi-layer perceptron-posteriors. Each feature stream is reduced by linear discriminant analysis to 45 components, including temporal context of size ± 4.

		Testing corpora (WER [%])			
	Feature type	dev10	eval10	eval09	dev09
MFCC	—	21.6	18.2	16.7	29.8
+ MLP-posteriors	MRASTA	20.6	17.0	15.7	28.2
	Hier-MRASTA	20.4	16.7	15.4	27.7

around 0.2-0.3% absolute in word error rate on the Spanish corpora. The first network of the hierarchy discriminates the targets using the most significant and important features. The second network distinguishes the targets further by taking into account less important information. As shown, providing the same information in the first network is masked by the most relevant information. Similar to the number of multiple layers, increasing the number of networks in the hierarchy leads from localized to global features.

4.2.2 Summary

This section analyzed the hierarchical processing framework trained on short-term and long-term features. As input for the next network in the hierarchy the previously trained posterior estimates as well as additional features were used.

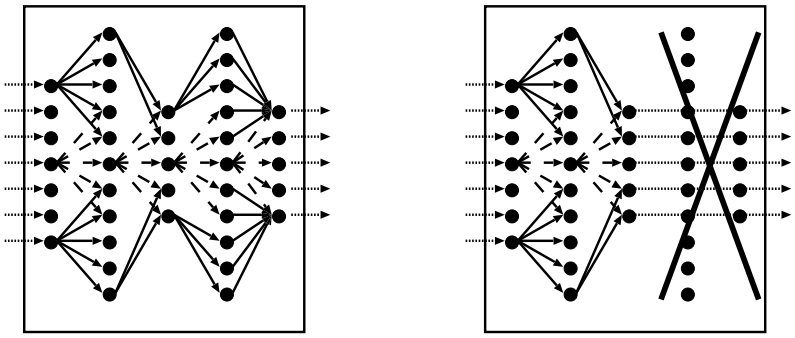
We verified the results presented in [Valente & Vepa$^+$ 07, Valente & Magimai-Doss$^+$ 09] where the fast and slow modulation frequencies of the multi-resolution RASTA processing were presented at different stages of the hierarchical processing. We showed that the ANN training benefits from splitting the features and the hierarchical framework on Spanish and Chinese.

4.3 Bottle-neck Processing

This section introduces the concept of probabilistic bottle-neck features derived from an ANN. Instead of using the activation of the output layer, the activation of an inner layer is taken into account. A general overview of the feature reduction technique using ANNs is presented in [Rumelhart & Hinton$^+$ 88] and in [Bishop 96, pp. 314 ff]. On an image task, [Hinton & Salakhutdinov 06] have shown that deep neural networks can learn much better low-dimensional representations than the principal component analysis.

The activation from a hidden layer is used for the first time in speech recognition in [Chen & Chang$^+$ 03] and [Chen & Zhu$^+$ 04]. The linear output of the large hidden layer is reduced by a merger multi-layer perceptron resulting in hidden activation temporal pattern based features. For the bottle-neck feature extraction this concept is modified by the following points:

- Increase the number of hidden layers



(a) Multi-layer perceptron during training

(b) Multi-layer perceptron during decoding

Figure 4.3 MLP-BN feature extraction. Different multi-layer perceptron structures are used in training (a) and decoding (b). During decoding all last layers after the bottle-neck are skipped. In the bottle-neck layer no activation function is applied.

- Introduce bottle-neck (narrow hidden layer)
- Output: linear activation of the bottle-neck layer

This modification results in the bottle-neck feature extraction method, described in detail in [Grézl & Karafiat$^+$ 07]. Figure 4.3 illustrates the concept of the bottle-neck feature extraction.

During training of the multi-layer perceptron a full 4-layer network is trained on the phonetic targets. In decoding all layers after the bottle-neck are skipped (here: the last two layers) and the ANN is reduced to a 2-layer network. The features obtained from the bottle-neck layer are referred to as probabilistic bottle-neck features. The goal in [Grézl & Karafiat$^+$ 07] has been to compress the input raw features in any arbitrary size and on the other hand to ensure a good class separability of the output features. Normally, the first hidden layer is large enough to provide the necessary modeling power. The last layer is again enlarged to further improve the classification error.

4.3.1 Experimental Results

We perform two main experiments to verify the results given in [Grézl & Karafiat$^+$ 07]. In the first experiments, bottle-neck features are extracted where the size of the bottle-neck is fixed to the number of phonetic targets. In the second experiment, the number of hidden nodes in the bottle-neck layer varies from 33 to 100. All experiments in this section are performed on the Spanish Quaero corpus (see Section A.3 for details). The tandem systems are trained on the linear discriminant analysis transformed MFCCs augmented by the probabilistic multi-layer perceptron based bottle-neck features. The bottle-neck features are normalized further by mean (μ) and variance (σ) and transformed by principal component analysis or linear discriminant analysis on 9 consecutive frames, thus including temporal context.

We keep the configurations of the hidden layers of the multi-layer perceptrons simple. The number of nodes in the first and last hidden layer is set to 4000 and 2000 respectively. The

Table 4.6 Comparison of different post processing steps of MLP-BN features on Quaero Spanish after speaker adaptation using SAT/CMLLR. The tandem systems are trained on MFCCs reduced by linear discriminant analysis augmented by the multi-layer perceptron probabilistic features. The MFCCs are reduced by linear discriminant analysis to 45 components, including a temporal size of ± 4 frames. The multi-layer perceptron is trained directly on the MFCCs.

	MLP post-processing		GHMM	Testing corpora (WER [%])			
Feature type	Transform	Context	Input size	dev10	eval10	eval09	dev09
MFCC	—	—	45	21.6	18.2	16.7	29.8
+ MLP-posteriors	PCA	± 0	68	20.7	17.0	15.6	28.6
	LDA	± 4	90	20.4	17.0	15.7	28.7
+ MLP-BN	NORM (μ, σ)	± 0	78	21.1	17.2	16.0	28.9
	PCA			20.5	16.8	15.6	28.5
			56	20.4	16.6	15.7	28.2
	LDA	± 4	90	20.3	16.5	15.4	27.7

bottle-neck is fixed to 33, 50, 75 and 100 nodes to analyze the significance of the bottle-neck size. A bottle-neck of size 33 allows a direct comparison between the 33 posterior features and the probabilistic features obtained from the bottle-neck. As input for the bottle-neck, 9 frames of the MFCCs are augmented with the first derivative and the first component of the second derivative, resulting in a 297 dimensional feature vector.

As shown in Table 4.6, we could verify the results given in [Grézl & Karafiat$^+$ 07]: the bottle-neck features improve the system performance. Even though the improvements on the development set of 2010 is small —0.1% absolute in word error rate— we obtain a better generalization of the features on all other corpora. On the evaluation corpora the systems are improved up to 0.5% absolute in word error rate.

Furthermore, a decorrelation step of the bottle-neck probabilistic features is necessary. When normalized bottle-neck features are used alone, the system performs worse compared to the system using the posterior estimates. Therefore, a principal component analysis or linear discriminant analysis transformation is required. The final feature size after principal component analysis plays a minor role. The principal component analysis transformed feature vector without feature reduction achieves similar results as the reduced features (11 dimensional). The reduced feature size is chosen to keep 95% of the variability of the feature vector.

In the second experiment we analyze the significance of the size of the bottle-neck. Table 4.7 summarizes the experimental results using a speaker adapted model. The influence of the size of the bottle-neck is negligible. Even if the difference on the development corpus of 2010 is significant, almost equal results are obtained on the other corpora. Nevertheless, the best performance is achieved using a bottle-neck of size 50 or 75. As shown in Section 7.5, the bottle-neck size depends on the input feature size. When the number of input feature is increased, the size of the bottle-neck has to be increased as well. The worst result is observed with a bottle-neck size of 33, but the log-posteriors are outperformed even with this configuration.

Table 4.7 Comparison of the influence of the size of the bottle-neck on Quaero Spanish. The tandem systems are trained on the MFCCs augmented by the multi-layer perceptron based bottle-neck (MLP-BN) features. Each feature stream is reduced to 45 components by linear discriminant analysis, including a temporal context of ± 4 frames. The systems are speaker adapted using SAT/CMLLR.

	MLP-BN	Testing corpora (WER [%])			
Feature type	Size	dev10	eval10	eval09	dev09
MFCC	—	21.6	18.2	16.7	29.8
+ MLP-BN	33	20.3	16.5	15.4	27.7
	50	19.9	16.3	15.2	27.8
	75	20.1	16.3	15.1	27.6
	100	20.1	16.3	15.1	27.5

4.3.2 Summary

In this section we introduced and analyzed the bottle-neck feature extraction. We showed that bottle-neck features improve the system performance but decorrelation of the bottle-neck features was important. Moreover, the generalization to unseen data was much better here than for the multi-layer perceptron based posterior estimates.

In a second experiment we analyzed the significance of the size of the bottle-neck layer in the multi-layer perceptron. We found that the size of the bottle-neck played a negligible role. Significant differences were only obtained on the development corpus, not on the other testing corpora.

Given the results mentioned above, the size of the bottle-neck will be fixed to the size of the phonetic targets trained on. This will allow a direct comparison of the posterior estimates and the bottle-neck features.

4.4 Hierarchical Bottle-neck Processing

In this section we introduce a new processing type which combines the advantages of the hierarchical ANN and the bottle-neck processing. As shown in Section 4.2 and Section 4.3, the hierarchical as well as the bottle-neck processing improve the baseline systems. Therefore, the hierarchical ANN training is modified in such a way that the features obtained from the networks use the bottle-neck framework. Each multi-layer perceptron in the hierarchy is trained using three hidden layers, whereas the linear activations from the bottle-neck (second hidden layer) are taken as features. Modifying the hierarchical framework in Figure 4.2 on page 61 by introducing the bottle-neck structure results in the hierarchical bottle-neck processing shown in Figure 4.4. We have successfully applied the hierarchical bottle-neck processing approach to an American and British English task [Plahl & Schlüter$^+$ 10]. In order to verify and compare the results given there, we present experimental results on Spanish and Chinese.

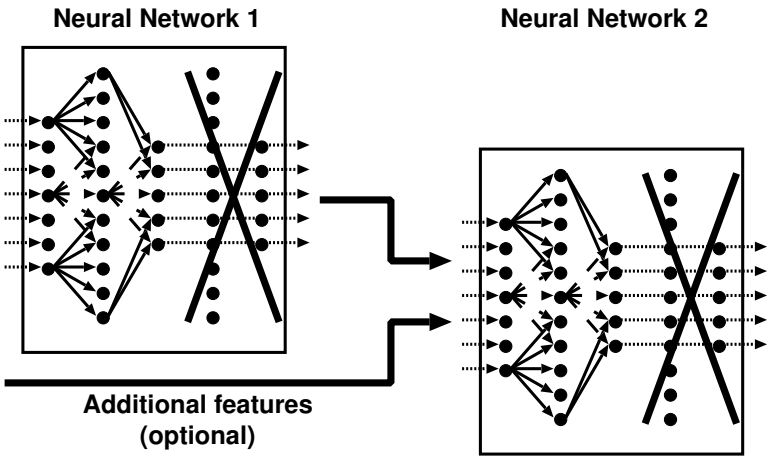


Figure 4.4 Hierarchical bottle-neck processing framework. The linear activations derived from an inner layer of a previously trained network are presented as input. During training, the full multi-layer perceptron is used. Optionally, the probabilistic features are augmented with any other feature stream.

Table 4.8 Multi-layer perceptron configurations of the hierarchical bottle-neck processing for Quaero Spanish. The fast (F-MRASTA) and slow (S-MRASTA) modulation frequencies of the multi-resolution RASTA processing are provided in different stages of the hierarchy.

	BN-MRASTA	Hier-BN-MRASTA		A-Hier-BN-MRASTA	
Configuration	NN	NN-1	NN-2	NN-1	NN-2
Input	MRASTA	F-MRASTA	NN-1 S-MRASTA	CRBE F-MRASTA	NN-1 CRBE S-MRASTA
Size	432	216	423	351	370
Hidden layer 1	4000	4000	4000	4000	4000
Hidden layer 2	33	33	33	33	33
Hidden layer 3	2000	2000	2000	2000	2000
Phonetic targets	33	33	33	33	33
MLP-BN transform	NORM	PCA	NORM	PCA	NORM
Final feature size	33	15	33	15	33

4.4.1 Experimental results

Experimental results performed on an American and British English task are already presented in [Plahl & Schlüter+ 10]. We have redone the experiments on Spanish and Chinese to be able to compare the results with other multi-layer perceptron feature extraction methods presented in this work.

Table 4.8 summarizes the configuration of the first and second multi-layer perceptron in the hierarchy. The final output size of the first network (NN-1) is chosen to keep 95% of the variability of the 33 bottle-neck features. Therefore, the principal component analysis transforms the bottle-neck features of NN-1 down to 15 components. The second network is trained on the slow modulation frequencies (S-MRASTA) and the principal component analysis reduced

Table 4.9 Comparison of hierarchical bottle-neck features on Quaero Spanish. The hierarchical processing divides the multi-resolution RASTA (MRASTA) features into fast and slow modulation frequencies (Hier-MRASTA). The A-Hier-MRASTA augments the input for the second network by critical band energies. The tandem systems are trained on the MFCCs reduced by linear discriminant analysis to 45 components and different MLP-BN features reduced by principal component analysis to 23 components. All systems are speaker adapted using SAT/CMLLR.

		Testing corpora (WER [%])			
	MLP feature type	dev10	eval10	eval09	dev09
MFCC	—	21.6	18.2	16.7	29.8
+ MLP-BN	MRASTA	21.0	17.0	16.1	28.0
	Hier-MRASTA	20.8	16.9	15.8	27.9
	A-Hier-MRASTA	20.6	16.8	15.5	27.7

features from the previous network (NN-1). The temporal context of the NN-1 features is increased by ± 4 frames. The slow modulation frequencies already include a long temporal context. No window is needed there.

After training the hierarchical ANNs we build tandem systems using MFCCs and the different probabilistic bottle-neck features derived from NN-2. Depending on the topology and input features used, the resulting features are named *BN-MRASTA*, no hierarchy, and *Hier-BN-MRASTA*, a cascade of two networks, and *A-Hier-BN-MRASTA*, hierarchy with additional features. In Table 4.9 the three features are referred to by the additional concepts used.

Spanish

We use the general Spanish setup of the acoustic model as described in Section A.3 to obtain the BN-MRASTA and Hier-BN-MRASTA and A-Hier-BN-MRASTA features. As mentioned, in the hierarchical processing the probabilistic bottle-neck features of NN-1 are reduced by principal component analysis to 15 components. The corresponding experimental results of the different features derived from the hierarchical bottle-neck framework are summarized in Table 4.9.

Even though the improvements obtained on Spanish are not large, we have successfully combined the hierarchical ANN processing and the bottle-neck structure. As for the hierarchical ANN experiments using the posterior estimates, an additional gain in the hierarchy is obtained by providing critical band energy features.

The small difference in the final recognition results on Spanish are mainly caused by the size of the bottle-neck. In Section 7.5 we will show that the quality of the bottle-neck features depends on the size of the bottle-neck as well as on the input feature size. Depending on the size of the input features the bottle-neck layer has to be enlarged.

Table 4.10 Comparison of hierarchical bottle-neck features on Gale Chinese. The hierarchical processing divides the multi-resolution RASTA features into fast and slow modulation frequencies (Hier-MRASTA). The A-Hier-MRASTA augments the input for the second network by critical band energies. The tandem systems are trained on the MFCCs and different multi-layer perceptron features. Each feature stream is reduced by linear discriminant analysis to 45 components, including a temporal size of ± 4 frames. All systems are speaker adapted using SAT/CMLLR.

		Testing corpora (WER [%])			
	MLP feature type	dev07	dev08	eval08	eval07-seq
MFCC	—	13.8	12.9	17.4	14.7
+ MLP-posteriors	MRASTA	13.4	13.0	17.1	14.4
	Hier-MRASTA	12.9	12.5	16.1	13.6
	A-Hier-MRASTA	12.6	12.2	16.0	13.6
+ MLP-BN	MRASTA	12.8	12.3	16.4	13.6
	Hier-MRASTA	12.1	11.6	15.4	13.1
	A-Hier-MRASTA	12.0	11.7	15.0	13.0

Chinese

The dependency of the bottle-neck size and the final recognition performance is verified on Chinese. The general configuration of the hierarchical bottle-neck processing framework for Chinese is similar to the setup for Spanish. Since the number of phonetic targets used to model the language is larger than for Spanish, the bottle-neck is increased. A principal component analysis reduces the 71 probabilistic bottle-neck features obtained from the first network of the hierarchy to 40 components. As mentioned before, the same size for the bottle-neck as the number of phonetic classes allows a direct comparison of the posterior and the probabilistic features. As for Spanish, we extract *BN-MRASTA* and *Hier-BN-MRASTA* and *A-Hier-BN-MRASTA* features. The tandem systems are trained on the probabilistic multi-layer perceptron features as well as on the MFCCs. Each feature stream is transformed independently by linear discriminant analysis including a temporal context of ± 4 frames. Overall, the input of the tandem systems consists of a $45 + 45 = 90$ dimensional feature vector.

The improvements obtained on Chinese by the bottle-neck features in the hierarchy are larger than the improvement by the posterior estimates, Table 4.10. In the hierarchical ANN framework the posterior estimates are improved by 4% relative, whereas the hierarchical bottle-neck features result in a 6% relative improvement. The system can be further improved by providing additional features in the hierarchical bottle-neck framework, e.g. critical band energies. This is common on all languages and corpora tested.

The improvements of the hierarchical bottle-neck processing do not depend on the systems trained. Table 4.11 summarizes the speaker independent recognition results. In the speaker independent case, larger relative improvements are obtained, than after speaker adaptation. This behavior is already observed in many other experiments of this work. The linear transformation in the speaker adaptation step models similar transformations which are performed by the hierarchical ANN as well. Nevertheless, the final hierarchical bottle-neck framework achieves significant improvement when the bottle-neck size is large enough to encode the input features.

Table 4.11 Comparison of hierarchical bottle-neck features on Gale Chinese. The hierarchical processing divides the multi-resolution RASTA features into fast and slow modulation frequencies (Hier-MRASTA). The A-Hier-MRASTA augments the input for the second network by critical band energies. The tandem systems are trained on the MFCCs and different multi-layer perceptron features. Each feature stream is reduced by linear discriminant analysis to 45 components, including a temporal size of ± 4 frames.

		Testing corpora (WER [%])			
	MLP feature type	dev07	dev08	eval08	eval07-seq
MFCC	—	16.3	14.8	20.1	16.7
+ MLP-posteriors	MRASTA	15.1	14.8	18.9	16.1
	Hier-MRASTA	14.2	13.9	17.9	15.1
	A-Hier-MRASTA	13.8	13.6	17.6	15.0
+ MLP-BN	MRASTA	14.2	14.0	18.1	15.2
	Hier-MRASTA	13.3	12.9	16.7	14.4
	A-Hier-MRASTA	13.0	12.7	16.2	14.0

4.4.2 Summary

In this section we developed and analyzed a new ANN topology. The new topology combined the hierarchical framework and the bottle-neck structure of an ANN. Each of the two approaches was optimizing different aspects of the network. The bottle-neck structure focused on a compact representation and the compression of the input data and the hierarchical ANNs improved the classification rate of confusable and currently not distinct classes. We showed that the hierarchical bottle-neck processing framework benefited from these different modeling and improved the overall performance of the ANN and the features derived from the ANN.

In addition, the size of the bottle-neck played an important role. When the bottle-neck was too small, compared to the input size of the network, the compression of the data in the bottle-neck layer as well as the generalization was poor. In this case, the bottle-neck structure benefited from the hierarchical ANN processing. Nevertheless, the improvements were much larger when a reasonable bottle-neck size was used.

CHAPTER 5

Recurrent Neural Networks

In this section we briefly describe the extension of the multi-layer perceptron concept to a *recurrent neural network (RNN)*. Instead of encoding information from previous time steps into the feature vector, RNNs use the output activations from a previous time step as input. These backward directed loops or recurrent connections of the RNNs behave like a memory block and encode the previous activations. In contrast to the multi-layer perceptron, any temporal expansion of the feature vector is not required any longer. This reduces the number of redundant parameters, which occur in the multi-layer perceptron.

This section is structured as follows: We start by introducing the concept of RNNs in Section 5.1 and explain the changes compared to the multi-layer perceptron. In Section 5.2 two approaches to train such RNNs are described. Since the temporal contextual information in an RNN is limited to the previous time steps, the concept of *bi-directional RNNs* is presented in Section 5.3. Bi-directional RNNs can fall back on the full input sequence during training and decoding. Next, we describe the *long-short-term-memory* structure in Section 5.4, which is especially designed by [Hochreiter & Schmidhuber 97] to cope with the problem of the vanishing gradient. The performance of the different RNN structures is evaluated in Section 5.5 on the Spanish task. Finally, we end this section with a short summary in Section 5.6.

5.1 Introduction

In the previous sections we obtain large improvements using posterior estimates or other probabilistic features derived from a multi-layer perceptron. The main disadvantage of this approach is the handling of temporal contextual information. Future and past temporal context of the current frame x_t are encoded into the feature vector $\hat{x}_t$, which increases the size of the input layer:

$$\hat{x}_t = (x_{t-\tau_1}, \cdots, x_{t-1}, x_t, x_{t+1}, \cdots, x_{t+\tau_2}), \tag{5.1}$$

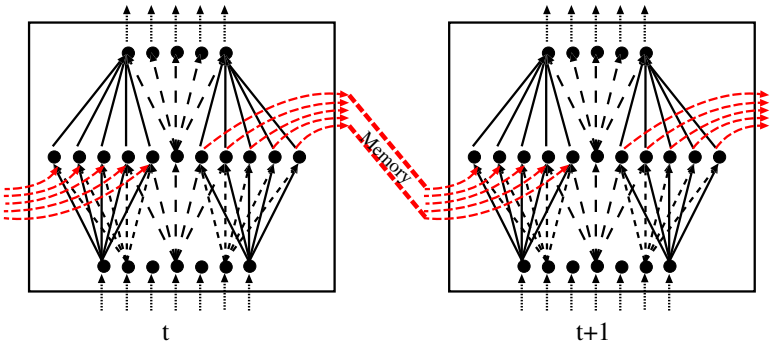


Figure 5.1 General structure of a 2-layer RNN. In a full connected RNN the hidden layer activations are looped backwards. The looped activations are buffered in a memory block and are available as input for all nodes within the same hidden layer. The looped activations or recurrent connections are marked by the dashed lines between the networks. The network shown here is unfolded in time.

Rather than model $p(s|x_t)$, the network depends on $\hat{x}_t$, and therefore we estimate $p(s|\hat{x}_t)$ for the output class s. In most cases the same number of past and future frames ($\tau_1 = \tau_2$) are taken into account, resulting in a symmetric window around the current frame x_t. One of the main issues of this workaround to include adjacent frames in the feature vector is that the number of past and future frames is fixed. Moreover, the feature vector expansion results in estimation of redundant weights during the multi-layer perceptron training.

In contrast to the results attained by encoding the past context into the feature vector, the output of the network can now be delayed by some time steps [Waibel & Hanazawa$^+$ 89]. In the implementation of such a time delay neural network, the past frames are provided by memory blocks. Nevertheless, the number of frames used in the input layer still stays the same. Moreover, the number of temporal contexts used in the input layer remains constant, too.

In most recognition tasks the dependency on the number of previous or future frames is unknown. When a window of a fixed size is used, the window is too small and relevant information is lost or the window is too large and the additional information causes confusion within the system. Providing an unknown number of previous frames to be used at a later time step leads to the concept of RNNs. In an RNN cyclical connections or backward directed loops exist between the output activation of the hidden layer and the input of any node within this layer. The previous frames are encoded in this recurrent connection and therefore, at least in theory, the whole past sequence can have an impact on the current output.

Depending on the integration of these recurrent connections into the network structure, different RNN models are known. *Hopfield* [Hopfield 82], and *Jordan* [Jordan 89], and *Elman* [Elman 90] networks are the most typical RNN representatives. Since the past sequence is encoded in the recurrent connections, each network has the advantage that the previous frames do not have to be encoded in the input vector any more. Nevertheless, the temporal context is limited to the past observation sequence. In order to provide future temporal context in the network, the output has to be delayed by several time steps. Figure 5.1 show the general concept of an RNN. The network is unfolded in time to visualize the temporal order.

The recurrent connections of the self-connected hidden layers are realized by feed-forward connections from the output activation of the hidden layer to the input of any node within the same layer. When the network is unfolded in time, the recurrent connections are treated as connection from time step t to the same hidden layer of time step $t+1$. Thus, the RNN introduces a simple way to include temporal context into the system without encoding the information in the input vector.

The forward pass of an RNN is similar to the forward pass of a multi-layer perceptron. In addition to the node activations of the previous layer, we have to keep track of the recurrent connections. The forward step becomes:

$$z_i^{(l)}(t) = \sum_j \underbrace{w_{ji}^{(l)} \cdot y_j^{(l-1)}(t)}_{feed-forward} + \sum_k \underbrace{w_{ki}^{(l+1)} \cdot y_k^{(l)}(t-1)}_{recurrent} \tag{5.2}$$

Note that the layer index (l) in the weights addresses the feed-forward connections, whereas the index $(l+1)$ symbolizes the recurrent connections.

5.2 Training

Several different training algorithms are developed to train RNNs. We will briefly describe the extension of the back-propagation algorithm to train the multi-layer perceptrons (Section 1.6.4) to the *back-propagation through time* algorithm and the *real time recurrent learning* algorithm. Details on each of the two learning algorithms are given in [Zell 94, Chapter 12].

5.2.1 Back-Propagation Through Time

The back-propagation through time algorithm [Werbos 90] is an extension of the back-propagation algorithm used to train multi-layer perceptrons. The recurrent connections of an RNN can be unfolded in time resulting in a network with feed-forward connections only. Repeating the application of the Chain rule results for node i in layer l, which is connected to other node within the layer (recurrent connections) as well as to the next layer (forward connections), in the error for time step $1 \leq t \leq T$, given by:

$$\delta_i^{(l)}(t) = \sigma_i'\left(z_i^{(l)}(t)\right) \cdot \left(\sum_j w_{ji}^{(l)} \cdot \delta_j^{(l)}(t+1) + \sum_k w_{ki}^{(l+1)} \cdot \delta_k^{(l+1)}(t)\right) \tag{5.3}$$

In order to distinguish the weights of the recurrent connections, the layer index is increased by 1. Since no future frames exist, the errors resulting from the time step $T+1$ are set to 0.

$$\delta_i^{(l)}(T+1) = 0, \forall i \tag{5.4}$$

Using this constraint, the errors $\delta_i^{(l)}(t)$ are calculated backwards in time, starting from time step T. After the error for each time step has been calculated, the update for a given weight w_{ij}^l

of layer l is summed up over time

$$\begin{aligned}\frac{\partial E_n}{\partial w_{ij}^{(l)}} &= \sum_{t=1}^{T} \frac{\partial E_n}{\partial z_i^{(l)}(t)} \cdot \frac{\partial z_i^{(l)}(t)}{\partial w_{ij}^{(l)}} \\ &= \sum_{t=1}^{T} \delta_i^{(l)}(t) \cdot y_j^{(l-1)}(t) \end{aligned} \tag{5.5}$$

The final update rule for the weights is not changed. In the implementation the weight update is modified to include the *momentum term* as described in Section 1.6.4.3.

5.2.2 Real Time Recurrent Learning

The real time recurrent learning algorithm to train RNNs is introduced by [Robinson & Fallside 87a, Robinson & Fallside 87b, Williams & Zipser 89]. In contrast to the back-propagation through time algorithm, the real time recurrent learning algorithm is applicable for online learning, since the weight updates are performed after each time step. Performing the updates after each time step results in memory requirements which are independent of the length of the observation sequence. The update of the weight w_{ij} of the current time step t corresponds to the errors w.r.t. the output nodes.

$$\begin{aligned}\Delta w_{ij}(t) &= -\eta \frac{\partial E_n(t)}{\partial w_{ij}} \\ &= -\eta \sum_{k=1}^{K} E_k(t) \cdot \frac{\partial y_k^{(L)}(t)}{\partial w_{ij}} \end{aligned} \tag{5.6}$$

The partial derivative of the activation $y_k^{(L)}(t)$ w.r.t. $w_{ij}^{(L)}$ for the current time step t is given by

$$\frac{\partial y_k^{(L)}}{\partial w_{ij}}(t) = \sigma'(z_k^{(L)}) \left(\sum_m w_{km} \frac{\partial y_m^{(L)}}{\partial w_{ij}}(t-1) + \delta(k,i) \cdot y_j^{(L)}(t-1) \right) \tag{5.7}$$

$\frac{\partial y_k^{(L)}}{\partial w_{ij}}$ is defined trough time, starting with the following constraint for time step $t = 0$:

$$\frac{\partial y_k^{(L)}}{\partial w_{ij}}(t=0) = 0 \tag{5.8}$$

In each time step the value of $\frac{\partial y_k^{(L)}}{\partial w_{ij}}(t)$ can be calculated and the weights are updated according to Equation (5.6).

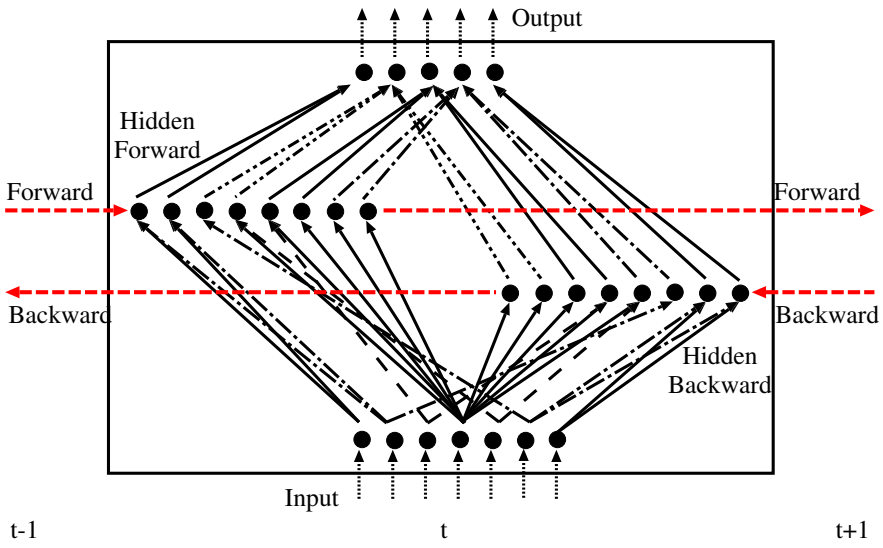


Figure 5.2 General structure of a 2-layer bi-directional RNN. The forward and backward RNNs are trained independently of each other. The hidden activations of the two networks are combined in a single output layer. The recurrent connections from the previous or from the next time steps are marked in red.

5.3 Bi-directional RNNs

As mentioned before, future temporal contextual information is provided by encoding the future context in the feature vector or by delaying the output by some time steps. Nevertheless, the numbers of future time steps are fixed. [Schuster & Paliwal 97] introduced the concept of *bi-directional RNNs* where two RNNs are trained. Figure 5.2 illustrates this concept of bi-directional RNNs. The first RNN is trained on the input sequence and the second network is trained on the reversed sequence. Whereas the first network copes with the temporal dynamics in forward direction, the second network is responsible for the dynamics in backward direction. In the output layer the sequence including the past context $x_1, \cdots, x_t$ and the sequence encoding the future context $x_T, \cdots, x_t$ are united. This leads to the full sequence $x_1, \cdots, x_T$ in the output layer to perform the class decision and to estimate the class posteriors.

$$p(s_k|x_t) = p(s_k|x_1, \cdots, x_T) \tag{5.9}$$

The general procedure for training and forwarding do not change much. Each network is decoded independently of each other. When the standard back-propagation through time algorithm is applied, the forward pass is executed first for each of the two networks. Since each network can be trained independently, the activations do not rely on each other. The output activations are calculated when the forward and backward activations for each time step are available. Afterwards, the errors are derived for each output and propagated back to the forward and backward directed networks. When all errors for all time steps are estimated, the final weight updates are derived and applied.

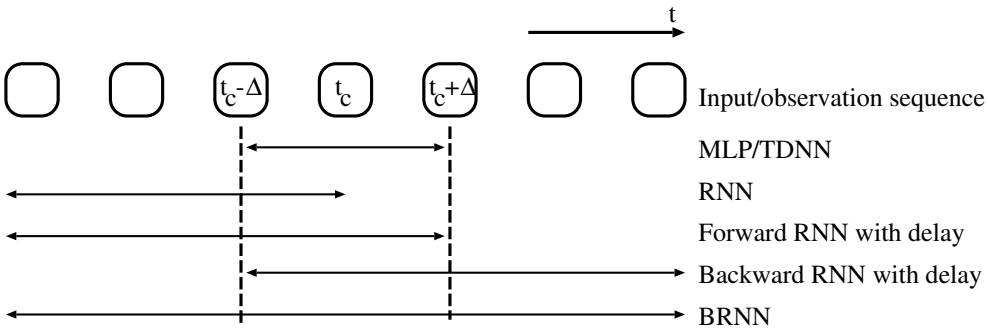


Figure 5.3 Visualization of the temporal contextual information used in different ANN topologies. The ANN covered are the multi-layer perceptron (MLP), the time delay neural network (TDNN) and the RNN with unidirectional or bi-directional structure. Depending on the topology, the context is encoded in the feature vector, is delayed, or given by the recurrent connections.

The training of a bi-directional RNN requires an update of the weights after presenting the full observation sequence. As in the back-propagation algorithm the error for back-propagation through time is observed after the full observation sequence has been presented. Furthermore, the training of bi-directional RNNs do not result in any large overhead, compared to classical RNNs.

Even though providing contextual information encoded in the feature vector or delaying the output improves the system performance, we will show in the experimental section that bi-directional RNNs outperform the standard RNN approaches. The training of the RNN benefits from the presence of all past and future frames at each time step. Nevertheless, the concept of long-short-term-memories has to be used to exploit the bi-directional structure best. The long-short-term-memory structure is explained in detail in the next section. Figure 5.3 summarizes the contextual information used in different ANN architectures used in this work.

5.4 Long-short-term-memory

This section gives a brief overview of the long-short-term-memory structure which is introduced by [Hochreiter & Schmidhuber 97]. Conventional RNNs, which use gradient descent based methods such as back-propagation through time or real time recurrent learning for weight parameter training, cannot model long-term dependencies due to the *vanishing gradient* problem. The temporal evolution of the error signal exponentially depends on the magnitude of the weights. Therefore, the back-propagated error either vanishes quickly or blows up as it cycles around the recurrent connections [Bengio & Simard$^+$ 94, Hochreiter & Schmidhuber 97]. The *long-short-term-memory RNN* overcome this problem. They enforce a constant error flow by introducing special cells.

5.4.1 Gating Nodes

Due to the vanishing gradient problem, classical RNNs, which are trained using the gradient descent algorithm, have difficulties to model long temporal dependencies in the observation sequence. Time lags greater than $5-10$ time steps seem to be hard to deal with for classical RNNs [Hochreiter & Schmidhuber 97]. The long-short-term-memory structure is specially designed to cope with the vanishing gradient problem. A recurrently self-connected linear node is added in the long-short-term-memory which is called the *constant error carousel* [Hochreiter & Schmidhuber 97, Hochreiter & Bengio$^+$ 01]. When the internal cell state is not changed, the constant error carousel back flow stays constant. Therefore, short-term as well as long-term dependencies up to 1000 time steps can be easily modeled by the long-short-term-memory structure [Hochreiter & Schmidhuber 97].

Each node in the hidden layer of an RNN is exchanged by the long-short-term-memory concept as illustrated in Figure 5.4. A long-short-term-memory node consists of a set of internal cells. The activation of the internal cell $c_j^{(l)}$ in layer l of unit j is controlled by three multiplicative gates: the input $I_j^{(l)}$, the forget $F_j^{(l)}$ and the output $O_j^{(l)}$ gate. The input of each of these gating units is the activation of the nodes of the previous layer and the activations obtained from the recurrent connections. Depending on the activation of the three gates, the internal cell is protected from irrelevant inputs and noise. When the input gate is closed or the activation of the gate is close to zero, new input to the long-short-term-memory node is blocked and the activation of the internal cell is not overwritten. When the output gate is open, the current activation of the cell is available for all other long-short-term-memory nodes within the network. The forget gate turns on and off the internal recurrent connections. In addition, the input and the output of the cells are squashed by symmetric sigmoid functions (g and h). Figure 5.4 illustrates the general structure of a long-short-term-memory node.

5.4.2 Training

Training of the long-short-term-memory RNN is performed similarly to the training of classical RNNs. Even though the first trainings are performed using a truncated version of the back-propagation through time algorithm, in the current implementation back-propagation through time and real time recurrent learning can be applied using the full past observation sequence.

The training of the long-short-term-memory nodes are divided again into the forward and the backward pass. In the forward pass the activations are calculated in a specific order. The order as well as the activation calculations of the different subunits in a long-short-term-memory node are given by Equation (5.10) to Equation (5.16). The net input, the input gate and the forget gate collect the input from the previous layer and from the recurrent connections. The linear activation of the cell state of the previous time step is considered as well.

Input gate:

$$I_i^{(l)}(t) = \sum_j w_{jI_i}^{(l)} \cdot y_j^{(l-1)}(t) + \sum_k w_{kI_i}^{(l+1)} \cdot y_k^{(l)}(t-1) + w_{c_iI_i}^{(l+1)} \cdot c_i^{(l)}(t-1) \tag{5.10}$$

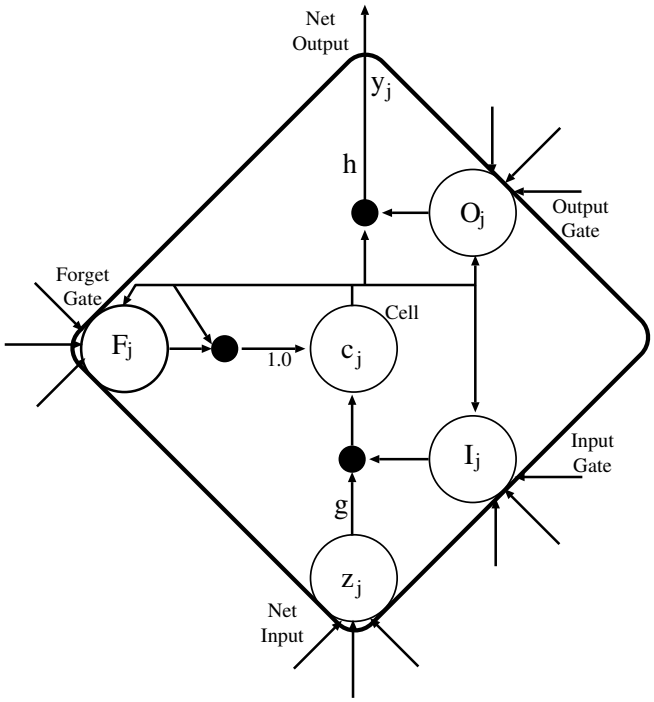


Figure 5.4 A long-short-term-memory node j with a recurrent self-connection of weight 1.0. The input and output and forget gates collect the input from all other long-short-term-memory nodes within the same layer (recurrent connections) and the previous layer (feed-forward connections). The inner cell state c_j is controlled by multiplicative units (dark circles). While the input and output gates scale the input and output of the cell, the forget gate scales the recurrent self-connection of the cell. Squashing functions (g and h) transforms the input as well as the output of the node.

Forget gate:

$$F_i^{(l)}(t) = \sum_j w_{jF_i}^{(l)} \cdot y_j^{(l-1)}(t) + \sum_k w_{kF_i}^{(l+1)} \cdot y_k^{(l)}(t-1) + w_{c_iF_i}^{(l+1)} \cdot c_i^{(l)}(t-1) \tag{5.11}$$

Net input:

$$z_{c_i}^{(l)}(t) = \sum_j w_{jc_i}^{(l)} \cdot y_j^{(l-1)}(t) + \sum_k w_{kc_i}^{(l+1)} \cdot y_k^{(l)}(t-1) \tag{5.12}$$

To determine the current activation of cell c_j, the previous cell activation and the net input are scaled by the activation of the forget gate and the input gate, respectively. When the activation of the output gate or input gate is close to 0, the influence of the previous cell activation or the net input is blocked.

Cell state:

$$c_i^{(l)}(t) = \sigma\left(F_i^{(l)}(t)\right) \cdot c_i^{(l)}(t-1) + \sigma\left(I_i^{(l)}(t)\right) \cdot g\left(z_{c_i}^{(l)}(t)\right) \tag{5.13}$$

where $\sigma(\cdot)$ is the activation function, a logistic sigmoid function in the range of $[0,1]$ (see Equation (1.17)) and $g(\cdot)$ a centered logistic functions in the range of $[-2,2]$.

$$g(x) = 4 \cdot \left(\frac{1}{1+e^{-x}} - \frac{1}{2} \right) = \frac{4}{1+e^{-x}} - 2 \tag{5.14}$$

After the cell state has been updated, the input activation of the output gate is derived by:

Output gate:

$$O_i^{(l)}(t) = \sum_j w_{jO_i}^{(l)} \cdot y_j^{(l-1)}(t) + \sum_k w_{kO_i}^{(l+1)} \cdot y_k^{(l)}(t-1) + w_{c_iO_i}^{(l+1)} \cdot c_i^{(l)}(t) \tag{5.15}$$

When the output gate is closed, the activation of the output gate is close to 0 and the output of the long-short-term-memory node scales to 0.

Net output:

$$y_i^{(l)}(t) = \sigma\left(O_i^{(l)}(t)\right) \cdot h\left(c_i^{(l)}(t)\right) \tag{5.16}$$

where $h(\cdot)$ is a centered logistic functions in the range of $[-1,1]$.

$$h(x) = 2 \cdot \left(\frac{1}{1+e^{-x}} - \frac{1}{2} \right) = \frac{2}{1+e^{-x}} - 1 \tag{5.17}$$

After the calculation of the forward pass, the corresponding errors and the error back flows are estimated. The partial derivatives of the output layer are computed as shown in Section 5.2. Finally, each weight w_{ij} is updated using the standard weight update rule from Section 1.6.4.3. As for the RNN, the momentum term is included in the update rule.

The error back flows of the different nodes in the long-short-term-memory are estimated using the following equations:

Net output:

$$\delta_{y_i}^{(l)}(t) = \sum_k w_{ki}^{(l)} \cdot \delta_k^{(l+1)}(t) + \sum_j w_{ji}^{(l+1)} \cdot \delta_j^{(l)}(t+1) \tag{5.18}$$

Output gate:

$$\delta_{O_i}^{(l)}(t) = \sigma'\left(z_{O_j}^{(l)}(t)\right) \cdot h\left(c_i^{(l)}(t)\right) \cdot \delta_{y_i}^{(l)}(t) \tag{5.19}$$

Cell state:

$$\begin{aligned} \delta_{c_i}^{(l)}(t) &= y_{O_j}^{(l)}(t) \cdot h'\left(c_i^{(l)}(t)\right) \cdot \delta_{c_i}^{(l)}(t) + y_{F_i}^{(l)}(t+1) \cdot \delta_{c_i}^{(l)}(t+1) \\ &\quad + w_{c_iI_i}^{(l)} \cdot \delta_{I_i}^{(l)}(t+1) + w_{c_iF_i}^{(l)} \cdot \delta_{F_i}^{(l)}(t+1) + w_{c_iO_i}^{(l)} \cdot \delta_{O_i}^{(l)}(t) \end{aligned} \tag{5.20}$$

Net input:

$$\delta_{z_i}^{(l)}(t) = y_{I_i}^{(l)}(t) \cdot g'\left(z_{c_i}^{(l)}\right) \cdot \delta_{c_i}^{(l)}(t) \tag{5.21}$$

Forget gate:

$$\delta_{F_i}^{(l)}(t) = \sigma'\left(z_{F_i}^{(l)}(t)\right) \cdot c_i^{(l)}(t-1) \cdot \delta_{c_j}^{(l)}(t) \tag{5.22}$$

Input gate:

$$\delta_{I_i}^{(l)}(t) = \sigma'\left(y_{I_i}^{(l)}(t)\right) \cdot g\left(y_{c_j}^{(l)}(t)\right) \cdot \delta_{c_j}^{(l)}(t) \tag{5.23}$$

5.5 Experimental Results

The experiments using RNN based posterior estimates are split into two major parts. In the first experiments we train several RNNs that differ only in their structure. The latter part investigates the significance of the temporal contextual information when bi-directional RNNs are used. Whereas the first part is performed on the full Spanish es-medium corpus, the second experiments are performed on the es-small corpus only.

5.5.1 Recurrent Neural Network Topologies

The experiments performed differ in the structure of the RNNs used. We train unidirectional and bi-directional RNNs in combination with and without the long-short-term-memory structure. Table 5.1 summarizes the recognition results of the four tandem systems trained. During the training of the different RNNs, the leaning rate is kept fixed. By adjusting the learning rate, corresponding to the performance on the validation set, we obtain a small but significant gain in performance.

The RNNs consist of one hidden layer and one output layer of size 33. The size of the hidden layer depends on the topology used. The simple full connected RNNs have a hidden layer size of 400, whereas the size of the hidden layer of the LSTM-RNN is decreased to 200. The hidden layer size is chosen to obtain a similar number of parameters. The unidirectional RNNs have about 200k parameters and the bi-directional RNNs have 400k parameters. As input for the training of the network we use the 16 dimensional MFCC features augmented by Δ and $\Delta\Delta_1$.

The final tandem system is trained on the MFCCs and the RNN based posteriors. Each feature stream is transformed independently by linear discriminant analysis and reduced to a 45-dimensional subspace using a temporal context of size ± 4. The acoustic model is adapted by speaker adaptive training using constrained maximum likelihood linear regression resulting in around 1.1M mixture densities in total.

Table 5.1 shows that the bi-directional RNNs outperform the unidirectional RNNs. Even though the difference is small for the conventional RNN, the bi-directional long-short-term-memory structure exploits the presence of the full input sequence best. Moreover, the bi-directional long-short-term-memory RNN benefits from the long-short-term-memory structure and outperform the unidirectional RNN and the bi-directional RNN. The best performance of RNN based posterior features is obtained when the long-short-term-memory structure is combined with the bi-directional approach. The performance of the bi-directional long-short-term-memory RNN improves further by adjusting the learning rate η. The improvements are about 0.4% absolute in word error rate on almost all testing corpora when the learning rate η

Table 5.1 Comparison of multi-layer perceptron and RNN based posteriors on Quaero Spanish after speaker adaptation using SAT/CMLLR. The different ANNs are trained on MFCCs using different context length. The final tandem systems are trained on MFCCs and the ANN posteriors. Each feature stream is reduced by linear discriminant analysis to 45 components, including a temporal context of ± 4 frames.

	ANN config		Testing corpora (WER [%])			
	Input	# weights	dev10	eval10	eval09	dev09
MFCC	—	—	21.6	18.2	16.7	29.8
+ MLP-posteriors	297	1.320k	20.4	16.9	15.5	28.4
+ RNN-posteriors	33	190k	21.3	17.4	16.0	29.0
+ bi-directional		380k	21.2	17.4	15.8	28.9
+ LSTM	33	195k	21.0	17.4	15.9	28.3
+ bi-directional		390k	19.8	16.3	15.0	26.8
+ adjust η			19.4	15.9	14.9	26.3

is modified according to the performance on the validation set. η is modified similarly to the modification of the learning rate for the multi-layer perceptron. Each time the difference in performance on the cross validation drops under 0.2% absolute, η is reduced by a factor of 2.

Compared to the multi-layer perceptron based experiments, the number of parameters in the network is decreased dramatically. In the RNN the temporal context is provided in the recurrent connection of the network, saving a large number of weight parameters. Nevertheless, the posteriors of an RNN do not always perform better than the multi-layer perceptron based posteriors. Although there is a large difference in the number of parameters, only the bi-directional bi-directional long-short-term-memory RNN outperforms the multi-layer perceptron result. The difference is around 1% absolute in word error rate resulting in a relative improvement of 5%. The number of parameters is reduced by a factor of 3. This result is quite impressive.

5.5.2 Temporal Context

In this experiment we analyze the influence of the temporal contextual information when the complete training sequence is available in any step during the training. As mentioned before, the experiments are performed on the es-small corpus containing 50 hours of audio data.

As shown in the experiments using multi-layer perceptron based posterior estimates, providing a temporal context of ± 4 frames results in a significant improvement during the training. The bi-directional long-short-term-memory RNNs are trained on the same temporal context as the multi-layer perceptrons. Therefore, the current input feature vector x_t is expanded resulting in the input feature vector $\tilde{x_t} = x_{t-4}, ..., x_t, ..., x_{t+4}$. The 2-layer bi-directional long-short-term-memory RNN consists of a hidden layer of size 128 and an input layer size of 16 or $16 \times 9 = 144$. The total parameter size is about 160K and increases to 300K when the context is encoded in the MFCC input feature vector. The linear discriminant analysis transformed posterior estimates of the RNN are augmented by the linear discriminant analysis transformed MFCCs to train a speaker adapted model using SAT/CMLLR. The corresponding multi-layer perceptron based feature tandem recognition systems use the multi-layer perceptron standard configuration with two hidden layers and a hidden layer size of 4000. Table 5.2 summarizes the RNN and

Table 5.2 Comparison of the influence of temporal context for training multi-layer perceptrons and RNNs. The context is encoded into the features vector, increasing the size of the input layer of the ANNs trained. The tandem system is trained on MFCCs augmented by the different ANN posteriors. Each feature stream is reduced by linear discriminant analysis to 45 components, including a temporal context of ± 4 frames. The acoustic model is speaker adapted using SAT/CMLLR.

	ANN input		Testing corpora (WER [%])			
	Context	Size	dev10	eval10	eval09	dev09
MFCC	—	—	22.3	18.5	17.0	30.6
+ MLP-posteriors	± 0	16	22.1	18.4	16.8	30.2
	± 4	144	21.1	17.7	16.3	29.3
+ BLSTM-RNN-posteriors	± 0	16	19.9	16.3	15.4	27.2
	± 4	144	20.1	16.4	15.3	27.1

multi-layer perceptron tandem recognition results after speaker adaptation.

As expected, encoding the temporal context into the feature vector does not lead to any significantly improvement when bi-directional long-short-term-memory RNNs are used. Moreover, the system using the expanded feature vector achieves almost the same performance as the system using just the current frame. Since the bi-directional networks provide the past and future context at each time step, the additional information encoded in the feature vector does not have any significant impact.

5.6 Summary

In this section we investigated RNNs and analyzed three different topologies. The bi-directional RNNs were able to provide the full sequence during the RNNs training, whereas the long-short-term-memory structure managed the vanishing gradient problem.

We showed that the number of temporal contextual information did not play any significant role when the full sequence was available. Nevertheless, the full sequence was exploited only in combination with the long-short-term-memory structure. This was understandable, since the vanishing gradient problem affects the usable context length in the classical unidirectional or bi-directional RNNs.

The best recognition performance was achieved by combining the long-short-term-memory approach and the bi-directional structure. Compared to the best multi-layer perceptron based posterior estimates on the same input features we obtained a 1% absolute improvement in recognition performance w.r.t. word error rate. Moreover, the bi-directional long-short-term-memory RNN trained reduces the number of parameters significantly by a factor of 3 and achieves a better word error rate than the corresponding multi-layer perceptron.

Due to the great performance of the bi-directional long-short-term-memory RNNs all further RNN based experiments will be performed using bi-directional long-short-term-memory RNNs. The learning rate will be adjusted similarly to the modifications of the learning rate during the multi-layer perceptron training.

CHAPTER 6

Domain and Language Portability of Neural Network based Features

In this section we analyze the effect of ANN based features trained on another language than the tandem system afterwards. In the previous section we showed that a large gain in performance is obtained when ANN based features are included in the speech recognition systems. All the investigations presented here, as well as other investigations, are extremely time consuming, whereas the decoding of an ANN is very efficient. Therefore, decoding a previously trained ANN to provide acoustic probabilistic features for tandem training and for recognition is an efficient reuse of available resources.

One possibility to save computational resources is discussed in Chapter 7 where several input feature streams are combined using different ANNs and ANN topologies. On the one hand, simplifying the development circle of the acoustic model reduces the computational costs significantly [Plahl & Schlüter$^+$ 11b]. On the other hand, the reuse of an already trained ANN is another simple and efficient way to save computational resources. In [Stolcke & Grézl$^+$ 06] short-term multi-layer perceptron posterior features, trained on 1800 hours of English data, improve the recognition results on an Arabic and a Chinese task. Yet, the best results in [Stolcke & Grézl$^+$ 06] are obtained when the multi-layer perceptron based features are trained on the same language as the acoustic model, even though a small amount of less than 100 hours are used to train both, the multi-layer perceptron and the acoustic model. The intra-lingual multi-layer perceptron Chinese or Arabic features seem to work best for [Stolcke & Grézl$^+$ 06].

When adapting the weights of a previously trained ANN to another language the ANNs benefit from a good initialization, e.g. when only a small amount of training data is available. Moreover, a good initialization of the ANN can save necessary resources during training [Hinton 02]. In [Tóth & Frankel$^+$ 08] a developed Hungarian system has been improved by adapting English trained PLP based multi-layer perceptron features to Hungarian, where 2000h of English data are provided to train the multi-layer perceptron. Again, a very small amount of intra-lingual data (7h) has been available only. Overall, a large amount of data of another language than for

the acoustic training is used for the ANN training, whereas the acoustic model is trained on a small amount of data only.

Even though each language has its own phoneme set, languages share phonetic distinction on the level of articulatory features —such as voicing, frication and nasality. This given, improvements obtained by cross-lingual features are expected, even if the ANN has never been trained on the specific languages or domain. This is underlined by [Qian & Xu+ 11] where different articulatory based features are used to provide multi-lingual features trained on a small amount of training data per language. In [Grézl & Karafiát+ 11] the phoneme inventory of several languages are unified to achieve a better generalization of the multi-layer perceptron based bottle-neck features. The general idea of a common phonetic database is not new. The development of a multi-lingual speech database has been the main goal of the GlobalPhone project resulting in the GlobalPhone database [Schultz 02].

The multi-layer perceptron features used in our approach are not especially developed for under resourced languages which is the main focus of the multi-layer perceptron based bottle-neck features in [Vu & Metze+ 12]. We used the hierarchical bottle-neck concept introduced in [Plahl & Schlüter+ 10] and presented in Section 4.4. The results presented here on the French database are the same as in [Plahl & Schlüter+ 11a]. For the investigation of the cross-lingual feature generation we concentrate on the following main aspects:

- The role of the structure/topology of the multi-layer perceptron—simple multi-layer perceptron topologies have been used so far,
- The degree of kinship of training and testing language for the (cross-lingual) multi-layer perceptron probabilistic feature extraction,
- The dependency on the amount of data used for training the multi-layer perceptrons.

Finally, we will show, whenever different intra- and cross-lingual ANN features are available without any extra costs, the systems trained on these features produce different complementary errors. Moreover, we gain slightly more by the combination of the different systems as shown in other system combination experiments on the same corpus.

6.1 Cross-lingual Feature Extraction

One of the main challenges in automatic speech recognition tasks is to simplify the developed methods and the system development circle. Moreover, optimizing the available computational resources without any loss in performance is hard work. Cross-lingual multi-layer perceptron based features have the ability to reuse a trained multi-layer perceptron on a different domain or language to provide multi-layer perceptron features for a given task.

In this section we briefly summarize the cross-lingual multi-layer perceptron based feature extraction method. While the training of a multi-layer perceptron is time and resource consuming, the decoding of the multi-layer perceptron is not. As mentioned before, languages share phonetic distinction at the level of articulatory features. Therefore, we concentrate on the following method to extract probabilistic cross-lingual multi-layer perceptron based features.

1. Train probabilistic features for a specific language ***E***
2. Keep the topology and the weights of the multi-layer perceptron fixed
3. Decode training and testing data for language ***F*** using the multi-layer perceptron trained on language ***E***.

As shown in the previous sections an exhausting investigation on different multi-layer perceptron topologies have been performed in the last years. The hierarchical bottle-neck topology (see Figure 4.4 on page 68) shows to give reasonable improvements over the other multi-layer perceptron topologies [Plahl & Schlüter$^+$ 10]. The hierarchical bottle-neck structure combines the advantages of the bottle-neck structure [Grézl & Karafiat$^+$ 07] as well as the advantages of the hierarchical ANN processing [Valente & Magimai-Doss$^+$ 09].

In the experiments presented here, two different hierarchical multi-layer perceptron topologies are chosen for training. The hierarchical ANNs are trained with and without the bottle-neck concept. This allows to analyze the influence of the phonetic targets of the specific source languages ***E*** on the target language ***F***. Moreover, the correlation between the bottle-neck concept and the data used for training could be analyzed, too. Thus allows a comparison experimental results already obtained on the same corpus.

6.2 Experimental Results

6.2.1 Cross-lingual Feature Extraction

Two main experiments are presented in this section. The first experiment has already been published in [Plahl & Schlüter$^+$ 11a]. There the cross-lingual feature experiments are performed on French and German and the multi-layer perceptrons are trained on English and Chinese. Since the results on French and German show the same effect, the German results are skipped as well as the results from using the multi-layer perceptrons trained on English. For comparison, we present the corresponding cross-lingual Chinese multi-layer perceptrons feature results on Spanish.

The general configuration of the multi-layer perceptrons for Chinese as well as for French or Spanish are not changed from the previous setups. In order to analyze the effect of the amount of training data used, the cn-small and the cn-large Chinese corpora are used for multi-layer perceptron training. Afterwards, the intra-lingual French multi-layer perceptron features and the cross-lingual Chinese multi-layer perceptron features are augmented by the linear discriminant analysis transformed MFCCs. The same configuration is used on Spanish. The multi-layer perceptron features themselves are transformed by principal component analysis to a final size of 30 components. A detailed description of the Chinese and French and Spanish systems as well as the corpora used are given in Appendix A. Whereas in the Chinese language the tonal information play an important role [Lei & Siu$^+$ 06], in these experiments the tonal information is skipped during the multi-layer perceptron and tandem training. We found that the tonal information does not help to improve the system performance of European languages.

Table 6.1 Comparison of cross-lingual and intra-lingual multi-layer perceptron features on Quaero French after speaker adaptation using SAT/CMLLR. The multi-layer perceptron trained on Chinese (CN) and French (FR) are used to produce the cross-lingual and intra-lingual multi-layer perceptron features. The different multi-layer perceptrons are trained using the hierarchical ANN (Hier-MRASTA) and the hierarchical multi-layer perceptron based bottle-neck framework (Hier-BN-MRASTA). The tandem systems are trained on MFCCs reduced by linear discriminant analysis to 45 components and multi-layer perceptron features reduced by principal component analysis to 30 components. The parameters are tuned on the development corpus, marked by *.

MLP training Language	MLP feature type			Testing corpora (WER [%])		
				dev10*	eval10	eval09
FR	(no MLP features)			24.1	25.4	34.2
	Hier-BN-MRASTA		(F2)	23.1	23.7	33.4
CN	Hier-MRASTA	230		23.7	24.3	33.6
	Hier-BN-MRASTA	230	(F3)	23.3	24.1	33.3
	Hier-MRASTA	1600	(F4)	23.1	24.1	33.1
	Hier-BN-MRASTA	1600	(F1)	22.4	23.5	32.7

Table 6.1 summarizes the experimental results of the intra- and cross-lingual multi-layer perceptron features on the French corpus. The systems marked by F3 and F4 include cross-lingual multi-layer perceptron features, but differ in the topology and the amount of training data used. Nevertheless, these two systems are competitive to each other as well as to the intra-lingual system F2. The relationship of training and testing languages is not relevant when the cross-lingual features are trained on a huge amount of data or the bottle-neck concept is included in the multi-layer perceptron training. When both, the huge amount of data and the bottle-neck concept are combined, the intra-lingual multi-layer perceptron features can be outperformed. Moreover, the bottle-neck structure produces language independent features and provides a good global structure of speech production tied over different languages. Even though the differences in the individual systems are smaller, we see the same effect on Spanish. Again, most of the gain obtained on the speaker independent model gets lost due to the speaker adaptation step. Table 6.2 summarizes the speaker adapted recognition results on the Spanish task.

The good performance of the cross-lingual multi-layer perceptron features is primarily related to the large amount of training data used, but also to the bottle-neck structure of the multi-layer perceptrons itself. Increasing the amount of training data used results in a more robust estimation of the weights of the multi-layer perceptron as well as the bottle-neck. The bottle-neck structure is not only relevant for a good class separability to provide a good and compact representation of the input features. The bottle-neck structure focuses on speech production aspects, common across different languages. This is supported by the fact that only the cross-lingual bottle-neck features gain over the intra-lingual features. Cross-domain and cross-system adaptation effects play an insignificant role. In [Plahl & Schlüter$^+$ 11a] we show the same effect on the English data base and for the German task.

6.2.2 Cross-lingual System Combination

In the previous experiments the performance is improved when cross-lingual multi-layer perceptron features are included in the system. To get a better analysis on how complementary the intra-lingual and cross-lingual systems are, we perform system combinations based on confusion networks as described in [Evermann & Woodland 00, Hoffmeister 11] for the French systems F1 and F2 and F3. The lattices are converted into confusion networks and the weights for the different systems are optimized on the development set of 2010. We combine two and three systems and the results are shown in Table 6.3. Other combinations like ROVER are tested as well, resulting in slightly worse results.

Not surprisingly, the best recognition results are obtained when all three systems are combined, whereas the influence of system F3 is pretty small. The worst combination result is observed when F1 and F3 are combined. This is also not surprising since both systems are trained on the cross-lingual multi-layer perceptron features on similar data and the same language. Therefore, these cross-lingual multi-layer perceptron based features produce similar errors. This is verified by the combination weight of each system. Whenever F1 and F3 are combined, the weight for F1 is dominant. When F2 and F3 are combined, the weights are equally distributed. The system combination results are summarized in Table 6.3.

On the basis of these results we conclude that the overall system can be simplified by just training a single system using the best multi-layer perceptron features. Nevertheless, most of the times more than one acoustic models are available. Combining systems F1 and F3 by system combination does not lead to any large improvements. The main reason is that F1 and F3 differ only in the amount of training data used. When the systems are combined, they should be as contrary as possible and competitive to each other at the same time.

Table 6.2 Comparison of cross-lingual and intra-lingual multi-layer perceptron features on Quaero Spanish after speaker adaptation using SAT/CMLLR. The multi-layer perceptron trained on Chinese (CN) and Spanish (ES) are used to produce the cross-lingual and intra-lingual multi-layer perceptron features. The different multi-layer perceptrons are trained using the hierarchical ANN (Hier-MRASTA) and the hierarchical multi-layer perceptron based bottle-neck framework (Hier-BN-MRASTA). The tandem systems are trained on MFCCs reduced by linear discriminant analysis to 45 components and multi-layer perceptron features reduced by principal component analysis to 30 components. The parameters are tuned on the development corpus, marked by *.

MLP training Language	MLP feature type	Testing corpora (WER [%])			
		dev10*	eval10	eval09	dev09
ES	(no MLP features)	21.6	18.2	16.7	29.8
	Hier-MRASTA	20.6	17.0	15.7	28.2
	Hier-BN-MRASTA	20.4	16.7	15.4	27.8
CN	Hier-MRASTA 230	20.9	17.1	15.9	28.3
	Hier-BN-MRASTA 230	20.7	16.9	15.7	28.1
	Hier-MRASTA 1600	21.0	16.8	15.8	27.9
	Hier-BN-MRASTA 1600	20.3	16.6	15.4	27.5

Table 6.3 Comparison of cross-lingual and intra-lingual system combination results for Quaero French (see Table 6.1 for details). The systems differ in the multi-layer perceptron probabilistic features used. The systems F1, F2 and F3 are combined by a frame wise lattice based system combination method. The systems used are marked by X.

Systems (MLP training language)			French (WER [%])		
F1 (CN)	F2 (FR)	F3 (CN.230h)	dev10*	eval10	eval09
X			22.4	23.5	32.7
	X		23.1	23.7	33.4
		X	23.1	24.1	33.1
X	X		21.4	22.4	31.8
X		X	22.0	22.9	32.2
	X	X	21.9	22.6	31.8
X	X	X	21.4	22.3	31.6

6.3 Summary

Overall, we developed a new method to optimize and simplify the training process and tested the method on French and Spanish. We showed that reusing previously trained multi-layer perceptrons leads to competitive recognition results depending on the topology used. Moreover, we outperformed the intra-lingual multi-layer perceptron features by using cross-lingual multi-layer perceptron features.

The performance of the cross-lingual features depended on the right topology of the multi-layer perceptron and the amount of training data used. Including just one of these aspects in the multi-layer perceptron training, the cross-lingual features achieved competitive results only. When combining both, the final tandem system benefited from the cross-lingual multi-layer perceptron and the degree of kinship between the two languages became less. As we showed, the cross-lingual multi-layer perceptron features even outperformed the intra-lingual multi-layer perceptron features.

Now, the training of complex multi-layer perceptrons for each language will no longer be required. In our case, the training of the hierarchical multi-layer perceptron based bottle-neck on the large Chinese corpus was sufficient to provide multi-layer perceptron based features which can be generalized to other languages as well. Depending on the structure of the other language, a huge difference was obtained. Although the difference got small, the errors produced by the cross-lingual systems differ from the intra-lingual systems. These error effects are efficiently exploited by system combination.

Overall, the system development circle can be simplified now without any loss of performance w.r.t. the word error rate. Instead of training ANN probabilistic features for each corresponding language within a project, a training of hierarchical multi-layer perceptron based bottle-neck features for one language —here Chinese— will be sufficient. Since the multi-layer perceptron feature extraction showed excellent efficiency for decoding, cross-lingual ANN features reduced the necessary amount of training resources and optimized the overall training and decoding process and the resources available.

CHAPTER 7

Neural Network Feature Combination

In conventional state-of-the-art automatic speech recognition systems a huge number of different acoustic short-term or long-term features are available. Several approaches are known which benefit from the information provided by different acoustic front-ends. The most promising method is system combination. The system combination approach has been proven to be superior to other feature combination methods [Zolnay 06]. Within the decoding framework system combination can be performed on different levels. Implemented in the adaptation step of the system it is referred to as *cross adaptation* and is proven to give considerable improvements [Guiliani & Brugnara 06]. Alternatively, lattice or N-best-list based system combination is applied to the final output of the individual systems [Evermann & Woodland 00]. [Hoffmeister 11] analyzes several system combination approaches and gives a detailed overview about the different methods. The main disadvantage of the system combination approaches is that the information of the different features is provided within the last step of the decoding framework. Therefore, a huge number of different systems have to be trained and decoded independently of each other resulting in high computational costs. Moreover, the best results are obtained only, when the systems are competitive to each other and as complementary as possible at the same time.

The combination of the (raw) features on feature level makes it necessary to train a single system only. The information of the different features is available during the training, which allows better discrimination and decisions. To reduce and to optimize the resources available, several approaches for combining acoustic features have been proposed in the last years. For example, in [Schlüter & Zolnay$^+$ 06] the combination is done explicitly on the feature level by linear discriminant analysis, though linear discriminant analysis has shown to be suboptimal [Zolnay & Schlüter$^+$ 05]. Furthermore, the combination in [Zolnay & Schlüter$^+$ 05] is done in an acoustic re-scoring framework. Even though both approaches achieve reasonable improvements, system combination seems to be superior [Zolnay 06].

The previous ANN experiments show that ANN features can provide complementary information to the final automatic speech recognition systems. As already mentioned in Section 4.2, additional features improve the performance of an ANN in the hierarchical framework. Therefore, it is obvious to use ANNs to perform feature combination. The main disadvantage of the linear discriminant analysis approach —linear dependencies could not be dealt with in a satisfying way— does not occur in the ANN approach.

Feature combination by ANNs is performed by augmenting the input feature streams and using the combined feature vector as input for training and decoding [Plahl & Schlüter$^+$ 11b]. One of the most important advantages of the ANN methods is the nonlinear transformation of the features. In most cases, the sigmoid activation function is used in the ANN training. We could benefit from the nonlinearity of the ANN to improve the combination of several feature streams. Another important fact concerns the computational costs. Whereas the input layer is enlarged to deal with the combined feature vector, all the other layers stay unchanged. Thus, the training and the decoding time of the ANN are increased insignificantly.

Chapter 3 and Chapter 5 have introduced different network types. We perform the neural network feature combination experiments on all network topologies shown above.

This chapter is structured as follows: We start reviewing the feature combination by linear discriminant analysis in Section 7.1. Section 7.2 discusses the combination of several features by simple multi-layer perceptrons, followed by the analysis of the system combination experiments in Section 7.3. Instead of combining the features, separate systems are trained on one feature stream and combined only afterwards. In Section 7.4 the structure of the multi-layer perceptron is changed into the hierarchical concept and to the bottle-neck concept, Section 7.5. We finish the neural network based feature combination with the combination results using RNNs in Section 7.6 and the hierarchical combination of RNNs and multi-layer perceptrons.

7.1 Linear Feature Combination

We now briefly reflect the method combining several features using linear discriminant analysis. As in [Plahl & Schlüter$^+$ 11b, Schlüter & Zolnay$^+$ 06], the experiments show that the linear discriminant analysis is suboptimal and not suitable for feature combination. On the one hand, strongly correlated acoustic features lead to degradation in word error rate due to unstable estimates in the projection matrix. On the other hand, a carefully pre-selection of features to be combined is necessary to avoid the performance degradation [Schlüter & Zolnay$^+$ 06].

In [Plahl & Schlüter$^+$ 11b] the experiments are performed on the Spanish es-small corpus. Here, we increase the amount of data and rerun the experiments on the Spanish es-medium corpus. The linear discriminant analysis combination results are independent of the amount of data used. Therefore, the same conclusions as in [Plahl & Schlüter$^+$ 11b, Schlüter & Zolnay$^+$ 06] are drawn.

We use the same approach to combine different short-term feature streams as in the ANN integration experiments (see Figure 3.2 (b) on page 35). A single linear discriminant analysis transformation is estimated to select the most relevant data from both feature streams. To cope with temporal context, the linear discriminant analysis estimation includes ± 4 consecutive

Table 7.1 Multiple feature combination results using linear discriminant analysis on Quaero Spanish. A single linear discriminant analysis matrix combines several feature sets, including a temporal context of ±4 frames. Furthermore, the systems are speaker adapted using SAT/CMLLR.

	GHMM	Testing corpora (WER [%])		
Feature type	Input size	dev10	eval10	eval09
PLP	45	23.0	19.6	17.8
GT		22.3	19.0	17.3
MFCC		22.3	18.5	17.0
+ PLP	60	22.2	18.7	17.0
+ MFCC		21.7	18.4	16.9

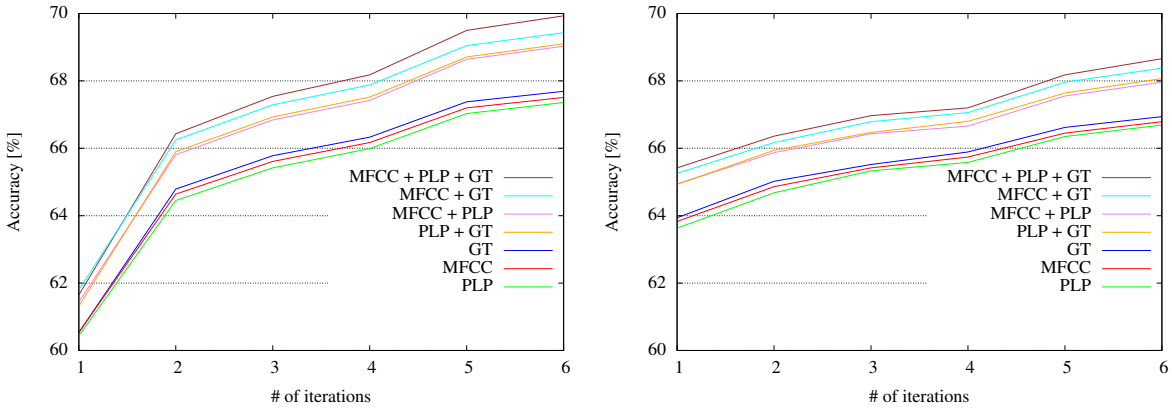
frames. Several final feature sizes are tried out and the best performance is obtained, when the linear discriminant analysis projects down to a 60 dimensional feature space.

As shown in Table 7.1, the feature combination by linear discriminant analysis improves significantly on the dev10 set only. Moreover, depending on the features combined a degradation of the recognition performance can be observed as well. This is due to the effect of numerical problems in the linear discriminant analysis estimation covered by [Schlüter & Zolnay$^+$ 06].

7.2 Single MLP Processing

As shown in the previous section, the linear discriminant analysis is suboptimal for feature combination. Therefore, we develop a new feature combination method based on multi-layer perceptrons. All input feature streams are augmented and the huge feature vector is presented as input for the ANN training. The main goal for the training of the ANN is to select the important feature components from the three different short-term feature streams. Each of the MFCC, PLP or GT feature streams in our experiments is augmented by its first derivative (Δ) and the first component of the second derivative ($\Delta\Delta_1$). In order to cope with temporal context, a window of ±4 frames is applied on top to obtain the final input feature vector.

Depending on the features combined, several multi-layer perceptrons are trained on the Quaero Spanish task and verify results on Chinese. Section A.3 gives detailed information of the Quaero Spanish corpus. For each language tandem systems are trained on the linear discriminant analysis transformed MFCC features and the multi-layer perceptron-posteriors, following the structure shown in Figure 3.2 (b). A linear discriminant analysis or principal component analysis transforms the multi-layer perceptron-posteriors further. The structure of the network used in the previous experiments is modified by changing the input layer size only. The number of nodes in the hidden layer and the output targets are kept. Figure 4.1 illustrates the two layer multi-layer perceptron concept. As usual, we use 4000 nodes in the hidden layer for Spanish as well as 33 phonetic targets. For Chinese, the output targets correspond to the 71 tonemes and 7500 nodes in the hidden layer.



(a) Training set **(b)** Validation set

Figure 7.1 Frame accuracy performance during the training of the multi-layer perceptrons. The accuracies are measured on the training and validation set. Overall, three different short-term features are combined.

7.2.1 Experimental Results on Spanish

The multi-layer perceptron posterior estimates trained on Quaero Spanish are transformed first by logarithm and by principal component analysis afterwards. These reduced features are combined with MFCCs transformed by linear discriminant analysis. The acoustic model is trained on a 68 dimensional feature vector. Table 7.2 summarizes the corresponding experimental results and Figure 7.1 show the progress on the training and cross validation set during the multi-layer perceptron training.

The performance of the accuracy on the training and cross validation set improves continuously when additional feature sets are provided. The gain from additional feature streams gets less when multiple feature streams are already combined. The best training performance is observed by starting with GT features and adding MFCCs as the second stream and finally a small gain is observed by including PLPs. The posterior estimates benefit from the different feature extractions and the different ways how these features represent the audio speech signal. Still, the three feature streams provide redundant information.

The progress in the frame accuracies of the multi-layer perceptron training shows the same tendency as the performance of the tandem systems. The system performance improves continuously when more feature streams are combined. Combining two feature streams results in 0.5% absolute improvement in word error rate on all testing corpora. The improvements get smaller when more feature streams are combined. Overall, the systems with multi-layer perceptron posteriors trained on just one single feature stream are improved by around 3.5% relative in word error rate.

Table 7.2 Combination of multiple feature sets using multi-layer perceptrons on Quaero Spanish. The tandem systems are trained on the MFCCs augmented by the MLP-posteriors. A linear discriminant analysis projects down the MFCCs to 45 components and a principal component analysis reduces the 33 MLP-posteriors to a size of 23. The acoustic model compensates speaker variations by adaption using SAT/CMLLR.

	MLP input feature		Testing corpora (WER [%])			
Feature type	Type	Size	dev10	eval10	eval09	dev09
MFCC	—	—	21.6	18.2	16.7	29.8
+ MLP-posteriors	MFCC	297	20.7	17.0	15.6	28.6
	GT		20.5	16.8	15.7	27.9
	PLP		20.5	17.0	15.6	28.6
	+ GT	576	20.0	16.8	15.7	28.4
	MFCC					
	+ GT	576	20.0	16.5	15.0	27.6
	+ PLP	594	20.0	16.6	15.3	27.8
	+ GT	873	19.8	16.3	15.0	27.5

Table 7.3 Combination of multiple feature sets using multi-layer perceptrons on Gale Chinese. The tandem systems are trained on the MFCCs augmented by the MLP-posteriors. Each feature stream is reduced by linear discriminant analysis to 45 components, including a temporal context of ± 4 frames. Speaker variations is compensated by speaker adapted using SAT/CMLLR.

	MLP input feature		Testing corpora (CER [%])			
Feature type	Type	Size	dev07	dev08	eval08	eval07-seq
MFCC	—	—	13.8	12.9	17.4	14.7
+ MLP-posteriors	MFCC	297	12.7	12.4	16.3	14.0
	GT		12.1	11.7	15.5	13.5
	PLP		12.6	12.2	16.2	13.7
	+ GT	576	11.5	11.2	14.9	12.8
	MFCC					
	+ GT	576	11.4	11.2	14.8	12.6
	+ PLP	594	12.0	11.6	15.2	13.1
	+ GT	873	11.2	10.8	14.6	12.4

7.2.2 Experimental Results on Chinese

The training of the Chinese tandem system is performed on the smallest of the three Chinese corpora containing 230h of speech (cn-small). Section A.1 gives detailed information on the Chinese system and the corpora used. The 90-dimensional input feature vector contains the linear discriminant analysis transformed MFCCs augmented by the tonal feature and the linear discriminant analysis transformed log-posterior estimates. Each feature stream includes temporal context of ± 4 frames.

As shown in the experiments for Spanish, the Chinese system benefits from the different feature streams combined in the multi-layer perceptron training. Table 7.3 summarizes the corresponding speaker adapted recognition results. Compared to the Spanish systems, the improvements obtained by multi-layer perceptron feature combination on Chinese are much more significant and noticeable. Moreover, the absolute improvements are similar to Spanish, but the relative improvements are much higher. Adding new feature streams during the multi-layer perceptron training results in a reduction of up to 5% relative in word error rate in the final tandem systems.

In the multi-layer perceptron feature combination approach, the number of final target classes plays an important role. Due to the tonal information in the Chinese language, the number of target classes is increased and the additional information from the contrary feature sets helps to distinguish the classes even further. The acoustic model trained on Chinese produces much lower error rates than the corresponding acoustic model on Spanish. Given this, the alignment obtained by the Chinese acoustic model is much better and small differences in the feature set can be efficiently used to discriminate the target classes.

7.2.3 Summary

In this section we developed a new feature combination method based on multi-layer perceptrons. The augmented feature streams were presented as input for the multi-layer perceptron training. The main advantage of the ANN combination approach was the ability

- to pick out the most relevant information,
- to cope with linear dependencies,
- to find a nonlinear transformation of the feature space,
- to encode the most relevant information in a small output vector

During training of the multi-layer perceptrons as well as in the final tandem system a significant gain in performance was observed. Due to the short-term feature used, the improvements became less when the number of additional feature sets was increased. This was not surprising, since only tiny changes were included in each feature extraction procedure. The largest gain was observed for the GT features, since the GTs were the features which differed most to the MFCC or PLP features.

The topology and the configuration of the multi-layer perceptrons did not change much. The increased feature vector affects the input size of the network, but all other layers stayed unchanged. Due to a larger input size, the training time was increased, but this was negligible to the overall training time and resources needed. When a full training of the multi-layer perceptron on all features will be a big issue, the following order should be taken: Start with GTs as a must have and add MFCCs to obtain the most relevant gain. The PLPs can be presented as input as well to achieve an additional gain.

7.3 System Combination vs. Feature Combination

System combination seems to be superior to the known feature combination methods, e.g. using a single linear discriminant analysis matrix [Zolnay & Schlüter$^+$ 05]. In this section we will show that this statement has to be revoked when an ANN combines different feature sets. The results presented here have been already published on a subset of the Spanish training corpus in [Plahl & Schlüter$^+$ 11b].

The system combination results presented in this thesis are performed by the lattice-based confusion network combination approach described in [Hoffmeister 11]. The approach is a further development of the confusion network combination approach introduced in [Evermann & Woodland 00]. Other system combination approaches described in [Hoffmeister 11] or ROVER [Fiscus 97] have been tried out as well but perform slightly worse.

In order to do a fair comparison of the feature combination results and the system combination experiments, we split the analysis into two main parts. In the first experiments, we compare the tandem systems trained on a single feature stream and system combination of the baseline systems trained on the same short-term feature set. The second part analyzes the ANN feature combination using multiple feature streams and system combination performed on the tandem systems using a single feature stream to train the multi-layer perceptron posteriors. As the system combination approach seems to be superior [Zolnay & Schlüter$^+$ 05] to the current feature combination approach, we do not expect any break through.

7.3.1 Combination of Single Stream Baseline Systems

System combination is performed by the confusion network combination approach [Hoffmeister 11, Evermann & Woodland 00]. Table 7.4 and Table 7.5 show the corresponding experimental results for Spanish and Chinese. The systems combined are marked by $\oplus$ of the features used to train the baseline system.

Many other publications (e.g. [Hillard & Hoffmeister$^+$ 07, Hoffmeister & Schlüter$^+$ 08, Willett & He 08, Sundermeyer & Nußbaum-Thom$^+$ 11]) show that system combination improves the overall performance and we achieve the best performance when all baseline systems are combined. Nevertheless, the tandem systems trained on the GT based multi-layer perceptron posterior estimates outperform the system combination results on all testing corpora for both Spanish and Chinese. The confusion network combination approach works best when the systems are competitive to each other, but different recognition errors occur. The information provided by the ANNs seems to produce similar effects. The final tandem system benefits from the MFCC feature stream as well as the multi-layer perceptron posterior estimates.

The tandem system is trained on the vocal tract length normalization warped MFCC features, whereas the baseline system is trained on MFCCs only. Nevertheless, training a vocal tract length normalization warped MFCC system for Spanish and combining the system with the PLP and/or GT baseline system results in an improvement of 0.2% absolute only. Overall, the tandem system outperforms each system combination result.

Table 7.4 Multiple feature set combinations using system combination on Quaero Spanish. The symbol $\oplus$ marks the system combination approach. Each system is trained on another short-term feature and is speaker adapted using SAT/CMLLR. The features are transformed by linear discriminant analysis, including a temporal context of ± 4 frames. In addition, the result of the tandem system based on MFCCs and multi-layer perceptron-posteriors are shown, where the multi-layer perceptron is trained on GT features.

	Spanish testing corpora (WER [%])			
Systems	dev10	eval10	eval09	dev09
MFCC	22.0	18.3	16.8	30.4
GT	21.7	18.2	16.6	29.2
PLP	22.2	18.5	17.0	29.5
$\oplus$ GT	20.9	17.3	16.1	27.8
MFCC				
$\oplus$ GT	20.8	17.2	15.8	28.2
$\oplus$ PLP	21.0	17.5	16.1	28.4
$\oplus$ GT	20.8	17.1	15.9	27.8
MFCC + MLP (GT)	20.5	16.8	15.7	27.9

Table 7.5 Multiple feature set combinations using system combination on Chinese. The symbol $\oplus$ marks the system combination approach. Each system is trained on another short-term feature and adapted by SAT/CMLLR to cope with speaker variations. The features are transformed by linear discriminant analysis, including a temporal context of ± 4 frames. In addition, the result of the tandem system based on MFCCs and multi-layer perceptron-posteriors are shown, where the multi-layer perceptron is trained on GT features.

	Chinese testing corpora (CER [%])			
Systems	dev07	dev08	eval08	eval07-seq
MFCC	14.1	12.6	17.3	14.4
GT	14.1	12.8	17.4	14.5
PLP	14.2	13.0	17.6	14.5
$\oplus$ GT	13.5	12.4	16.6	13.8
MFCC				
$\oplus$ GT	13.6	12.3	16.6	14.1
$\oplus$ PLP	13.6	12.4	16.5	13.9
$\oplus$ GT	13.3	12.3	16.4	13.8
MFCC + MLP (GT)	12.1	11.7	15.5	13.5

Table 7.6 Multiple multi-layer perceptron based feature combinations using system combination on Quaero Spanish. Each multi-layer perceptron is based on another short-term feature set. The tandem systems are trained on the MLP-posteriors augmented by MFCCs and are adapted using SAT/CMLLR. The symbol $\oplus$ marks the system combination approach and $+$ the multi-layer perceptron feature combination.

Systems	Spanish testing corpora (WER [%])			
	dev10	eval10	eval09	dev09
MFCC	20.7	17.0	15.6	28.6
GT	20.5	16.8	15.7	27.9
PLP	20.5	17.0	15.6	28.6
$\oplus$ GT	19.8	16.3	14.9	26.9
MFCC				
$\oplus$ GT	19.6	16.2	15.1	27.0
$\oplus$ PLP	19.7	16.4	15.0	27.2
$\oplus$ GT	19.5	16.2	14.9	26.9
MFCC + PLP + GT	19.8	16.3	15.0	27.5

7.3.2 ANN Posterior Tandem System Combination

As shown above, the system combination approach does not outperform the feature combination approach any longer. Since discriminative information encoded in the multi-layer perceptron features are included in the tandem system but not in the baseline system, we have conducted a second experiment. The tandem systems trained on multi-layer perceptron posteriors based on a single feature stream are combined and compared to the multiple feature multi-layer perceptron combination system. We ensure that all the information encoded in the multi-layer perceptron features as well as in the baseline MFCCs are provided both for system combination as well as for the multi-layer perceptron feature combination approach.

The training configuration of the acoustic model in the tandem system stays unchanged. All models use the same setup resulting in 1M Gaussian mixture densities. Details of the acoustic model are given in Section A.1 and Section A.3 for Chinese and Spanish respectively.

Table 7.6 and Table 7.7 summarize the system combination results on the tandem systems for Spanish and Chinese. Again, system combination improves the system performance but, as we have observed in the previous section, system combination is no longer superior to feature combination. Whereas on the Spanish task the performance difference is small, on Chinese the system combination is around 0.5% absolute worse compared to the best multi-layer perceptron feature combination result. Overall, we verified the results presented in [Plahl & Schlüter$^+$ 11b] on two languages with a larger amount of data used.

Table 7.7 Multiple multi-layer perceptron based feature combinations using system combination on Quaero Chinese. Each multi-layer perceptron is based on another short-term feature set. The tandem systems are trained on the MLP-posteriors augmented by MFCCs and are speaker adapted using SAT/CMLLR. The symbol $\oplus$ marks the system combination approach and $+$ the multi-layer perceptron feature combination.

	Chinese testing corpora (CER [%])			
Systems	dev07	dev08	eval08	eval07-seq
MFCC	12.7	12.4	16.3	14.0
GT	12.1	11.7	15.5	13.5
PLP	12.6	12.2	16.2	13.7
$\oplus$ GT	11.8	11.6	15.3	13.2
MFCC				
$\oplus$ GT	12.2	12.0	15.7	13.5
$\oplus$ PLP	11.8	11.7	15.3	13.3
$\oplus$ GT	11.7	11.5	15.1	13.1
MFCC + PLP + GT	11.2	10.8	14.6	12.4

7.3.3 Summary

In this section we compared the feature combination approach using multi-layer perceptrons and the system combination method on a Spanish and a Chinese task. We showed that the multi-layer perceptron based feature combination approach is not only competitive to system combination, but even outperforms the system combination results. Since the system combination was superior before, this result is a break through.

In order to achieve the best performance using different acoustic front-ends, systems based on a single front-end were trained. These systems were combined afterwards. The best system combination results were achieved when the systems were competitive and as contrastive as possible at the same time, which will be hard to realize. Moreover, training of several systems to be combined was time and resource consuming. The resources used depended on the complexity of the acoustic model.

Even though all feature streams had to be presented as input to train the multi-layer perceptron, the overall training time and the resources used were not increased much. Here, the input layer of the multi-layer perceptron had to be enlarged only. The training of all the different systems needed for system combination was avoided by combining the different acoustic features by a multi-layer perceptron. Overall, when different acoustic features will be available, feature combination using multi-layer perceptrons will be more reliable to train several systems as combining these systems afterwards by system combination.

7.4 Hierarchical MLP Feature Combination

As shown in Section 4.2, the hierarchical processing of multi-layer perceptrons improves the frame accuracy of the multi-layer perceptron training as well as the performance of the final tandem system. In this section we develop a new feature combination method by combining

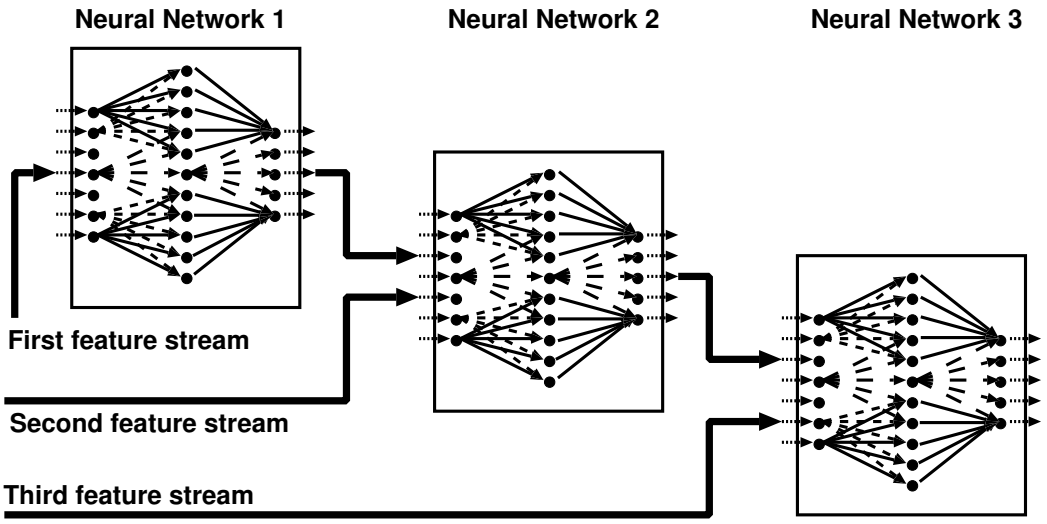


Figure 7.2 Hierarchical ANN feature combination setup to combine three feature streams. The output of a previously trained network is augmented by a new feature stream. Therefore, the next multi-layer perceptron is trained on the combined feature stream.

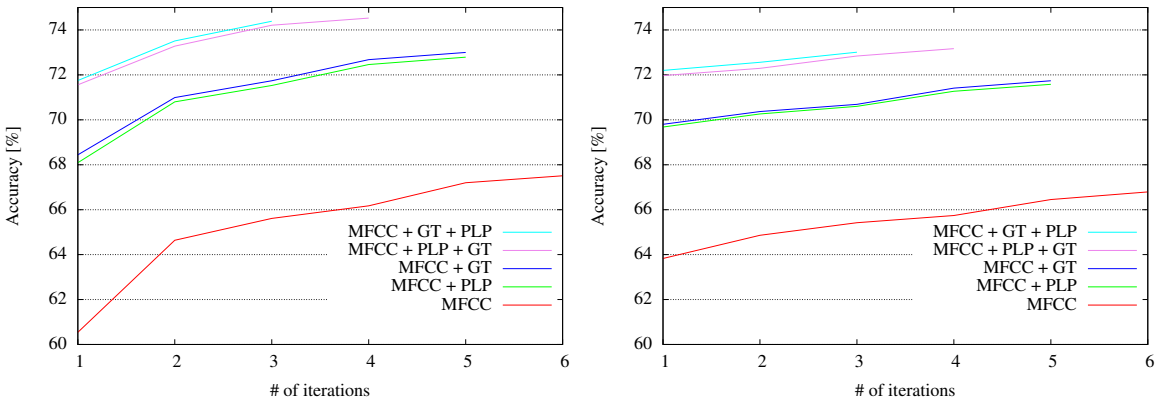
the hierarchical processing and the multi-layer perceptron feature combination method. Therefore, the individual feature streams are provided for multi-layer perceptron training in different stages of a cascade of multiple multi-layer perceptrons. We analyze the influence of the hierarchical processing when different feature sets are combined by multi-layer perceptrons during the network training and in the final tandem system.

The topology used in the hierarchical feature combination method follows the hierarchical processing with additional features presented in Section 4.2. Each feature stream is presented only once in the multi-layer perceptron cascade. Hence, in each stage of the hierarchical processing a new feature stream is added to the training. Figure 4.2 has to be modified by providing a new feature stream for combination in each stage of the hierarchy. The number of networks trained in the hierarchy corresponds to the number of feature streams used. Figure 7.2 illustrates the final hierarchical ANN feature combination topology.

7.4.1 Experimental Results

The experiments are performed on the es-medium Spanish task described in detail in Section A.3. The setup of the multi-layer perceptrons trained is kept as simple as in the previous experiments. In the 2-layer multi-layer perceptron, the hidden layer and the output layer contain 4000 and 33 nodes, respectively. The log-posteriors derived from NN-1 or NN-2 are transformed by logarithm and reduced by principal component analysis to 23 components before the next network is trained. Independently of the posterior estimates taken, the final tandem systems are based on the log phoneme posteriors augmented by the linear discriminant analysis transformed MFCCs.

The progress of the frame accuracy during the training of the multi-layer perceptrons is



(a) Training set **(b)** Validation set

Figure 7.3 Frame accuracies during the training of the hierarchical ANNs. The accuracies are measured on the training and validation set. Different short-term features and their combination are used as input to the multi-layer perceptron.

Table 7.8 Hierarchical ANN feature combination of different short-term features on Quaero Spanish. The features are provided in different stages of the hierarchy. The tandem systems are based on MFCCs augmented by the MLP-posteriors and is speaker adapted using SAT/CMLLR. A linear discriminant analysis reduces each feature stream to 45 components, including a temporal context of ± 4 frames.

Feature type	Hierarchical ANN Input feature	Testing corpora (word error rate [%]) dev10	eval10	eval09	dev09
MFCC	—	21.6	18.2	16.7	29.8
+ MLP-posteriors	PLP	20.6	16.9	15.4	28.1
	GT	20.4	16.8	15.7	27.8
	MFCC	20.4	16.9	15.5	28.4
	+ PLP	20.0	16.6	15.4	27.7
	+ GT	19.8	16.5	15.2	27.6
	+ GT	19.9	16.4	15.1	27.6
	+ PLP	20.0	16.4	15.1	27.5

summarized in Figure 7.3. As expected on the basis of the hierarchical processing, the frame accuracies on the training and validation set increase when different feature sets are combined. Even though different short-term features are provided in the second or third stage of the hierarchy, the overall performances in each stage are equivalent to each other. This leads to the conclusion that the type of feature presented is not significant. More importantly, the additional features provide complementary information to the current posterior estimates obtained by the previous multi-layer perceptron.

Table 7.8 summarizes the corresponding experimental tandem recognition results. The results after speaker adaptation show a similar performance as the frame accuracy results in Figure 7.3 intended. Even though the performance differences of the individual tandem systems

Table 7.9 Comparison of multi-layer perceptron based feature combination using a single network or the hierarchical framework on Quaero Spanish. The tandem systems are trained on the MFCCs and the different multi-layer perceptron based posteriors. Each feature stream is reduced by linear discriminant analysis to 45 components, including a temporal context of ± 4 frames. The augmented features are speaker adapted using SAT/CMLLR.

		Testing corpora (WER [%])			
	Feature type	dev10	eval10	eval09	dev09
Single MLP	MFCC	20.4	16.9	15.5	28.4
	+ PLP	20.1	16.6	15.3	27.7
	+ GT	19.7	16.3	15.0	27.5
Hierarchical ANN	MFCC	20.4	16.9	15.5	28.4
	+ PLP	20.0	16.6	15.4	27.7
	+ GT	19.8	16.5	15.2	27.6

w.r.t. the word error rate are not large, the hierarchical structure of the multi-layer perceptron improves the overall performance of the tandem systems as well. Nevertheless, one major difference of the frame accuracy results and the tandem recognition results exits: Whereas the order of the features in the hierarchy does not play any role during the training of the multi-layer perceptrons, the final tandem systems are sensitive to the chosen order. The tandem system trained on the posteriors of the hierarchical order MFCC-PLP-GT improves the system performance in each step, the hierarchical MFCC-GT-PLP show weaker performance compared to the MFCC-GT tandem system. This could be explained by the worse performance of the PLP features compared to the other short-term features and overfitting to the given data.

7.4.2 Hierarchical Combination vs. Single Network Combination

As shown in the experimental section, the hierarchical ANN feature combination approach improves the system performance by providing the different features at different stages of the hierarchy. In Section 7.2 we perform the combination of several short-term features by just training a single multi-layer perceptron on the combined feature vector. The corresponding results are shown in Table 7.2 on page 95. For an easier comparison of the two developed multi-layer perceptron combination methods, Table 7.9 summarizes the corresponding results.

As shown, each combination method achieves reasonable improvements on its own. Nevertheless, the feature combination does not benefit from the hierarchical structure. In all experiments the posterior estimates obtained by the single multi-layer perceptron feature combination method performs slightly better than the corresponding posterior obtained by the hierarchical framework. The gain is slightly increased if more feature streams are combined. Moreover, the superiority of the single multi-layer perceptron feature combination increases when tandem systems are trained on the posterior estimates only. The corresponding results are skipped. The frame accuracies show a different behavior: the frame accuracies during on the training and validation set increase a lot.

7.4.3 Summary

In this section we introduced a new multi-layer perceptron feature combination method by taking advantage of the hierarchical ANN processing framework. We made use of the multiple feature streams in different stages of the multi-layer perceptron cascade. The frame accuracy of the multi-layer perceptron as well as the final tandem system benefited from providing the different feature streams in the hierarchical structure of the multi-layer perceptron.

Compared to the single multi-layer perceptron feature combination method, the frame accuracies obtained by the hierarchical processing were much higher. Nevertheless, in the final tandem recognition system the same performance w.r.t. the character error rate or word error rate was achieved. This leads to the conclusion that even though the frame accuracies were improved, the final tandem system did not benefit from the improved posterior estimates. Again, the frame accuracy was one indicator for the final system performance but it was indispensable to perform a complete training of the final system to judge the given method.

Overall, the single multi-layer perceptron feature combination approach should be preferred, since it needed less training time and resources and the network is less complex.

7.5 Bottle-neck Feature Combination

Section 4.3 analyzes the behavior of a bottle-neck in the multi-layer perceptron topology and verifies the improvements obtained by the bottle-neck presented in [Grézl & Karafiat$^+$ 07]. In this coming section we modify our current feature combination setup by combining the bottle-neck structure of the multi-layer perceptron and the feature combination method developed in the previous sections.

The topology of the multi-layer perceptron is changed as follows. The training of a two-layer multi-layer perceptron for several feature streams (Section 7.2) is exchanged by training a 4-layer multi-layer perceptron on these different feature streams. The four-layer multi-layer perceptron includes the bottle-neck in the second hidden layer. In the decoding of the network, the linear activation of this bottle-neck is estimated. These final probabilistic features are taken as input feature for the tandem training.

To analyze the effect of the bottle-neck in combination with the feature combination method we perform different experiments. In the first experiments, the bottle-neck is fixed to 33 nodes as done in the bottle-neck experiments in Section 4.3. Depending on the number of feature sets combined and the input size, the performance of the bottle-neck starts to drop. Therefore, in the second experiment, we investigate the dependency of the input feature size and the size of the bottle-neck.

7.5.1 Small Bottle-neck Feature Combination

For the multiple feature combination experiment the single multi-layer perceptron topology in Figure 4.1 (a) is exchanged by the bottle-neck concept shown in Figure 4.3. During training the full network is trained whereas in decoding the last layers are skipped and the linear output of the bottle-neck is used as input.

Table 7.10 Effect of the bottle-neck processing for multi-layer perceptron based feature combination on Quaero Spanish. The tandem systems are trained on MFCCs augmented by the MLP-BN probabilistic features and speaker adapted using SAT/CMLLR. Each feature stream is reduced by linear discriminant analysis to 45 components, including a temporal context of ± 4 frames.

	MLP input feature		Testing corpora (WER [%])			
Feature type	Type	Size	dev10	eval10	eval09	dev09
MFCC	—	—	21.6	18.2	16.7	29.8
+ MLP-BN	MFCC	297	20.3	16.5	15.4	27.7
	GT	279	20.0	16.7	15.4	27.1
	PLP	297	20.2	16.6	15.4	27.7
	+ GT	576	19.6	16.2	15.1	26.7
	MFCC					
	+ GT	576	19.8	16.4	15.1	26.7
	+ PLP	594	19.8	16.2	15.0	27.2
	+ GT	873	20.0	16.6	15.4	27.4

We perform the training of the multi-layer perceptrons as well as the final tandem training on the Spanish es-medium corpus. A speaker adapted and a speaker independent tandem system are trained on the different probabilistic features. The Spanish development and evaluation data of 2010 and 2009 are used for decoding. Moreover, the parameters have been tuned on the development data of 2010 (dev10). The configuration of the multi-layer perceptron as well as the setup of the tandem system trained is similar to the system used in the previous sections.

The size of the nodes in the hidden layers are fixed to 4000, 33 and 2000 for the first, second and third hidden layer respectively. The number of input features varies from 300 to 900 depending on the number of feature streams and the feature type used. In the final layer the 33 phonetic classes of the Spanish language are presented. As input for the tandem systems the linear discriminant analysis transformed probabilistic bottle-neck multi-layer perceptron features are augmented by the linear discriminant analysis transformed MFCCs. Each linear discriminant analysis transformation includes a temporal context of ± 4 frames.

Table 7.10 summarizes the bottle-neck feature combination experiments. When two feature sets are combined, the resulting tandem system benefits from the additional feature source. The improvements are independent of the feature sets or the corpus used. When all three feature sets are combined, the result looks differently. The performance decreases on all corpora. As we will show in the next section, the size of the bottle-neck plays an important role when several feature sets are combined. When the size of the bottle-neck is very small compared to the input size, important and necessary information cannot be encoded in the bottle-neck. To achieve reasonable improvements, the bottle-neck size has to be increased.

Table 7.11 Effect of the bottle-neck size for multi-layer perceptron based feature combination on Quaero Spanish. The bottle-neck varies from 33 to 100. The tandem systems are trained on MFCCs augmented by the MLP-BN probabilistic features and speaker adapted using SAT/CMLLR. Each feature stream is reduced by linear discriminant analysis to 45 components, including a temporal context of ± 4 frames.

	MLP input feature		MLP-BN	Testing corpora (WER [%])			
Feature type	Type	Size	Size	dev10	eval10	eval09	dev09
MFCC	—	—	—	21.6	18.2	16.7	29.8
+ MLP-BN	MFCC	297	33	20.3	16.5	15.4	27.7
			50	19.9	16.3	15.2	27.8
			75	20.1	16.3	15.1	27.6
			100	20.1	16.3	15.1	27.5
	+ PLP	594	33	19.8	16.2	15.0	27.2
			50	19.7	16.1	14.9	27.0
			75	19.8	16.1	14.8	26.8
			100	19.6	16.2	14.9	26.8
	+ GT	873	33	20.0	16.6	15.4	27.4
			50	19.5	16.0	14.9	26.6
			75	19.2	15.9	14.8	26.4
			100	19.5	16.0	15.1	26.6

7.5.2 Dependency on the Bottle-neck Size

As shown in the previous section, the performance is increased when the size of the bottle-neck is too small compared to the input feature size. Therefore, it is necessary to increase the bottle-neck size to benefit from additional feature streams.

In the following experiments we enlarge the bottle-neck size from 33 to 50, 75 and 100. All other configurations of the multi-layer perceptron as well of the tandem system stay unchanged. The experiments in Table 7.10 of the previous section show no significant differences in which order the features have to be combined. Therefore, we start our experiments here with MFCCs and add the PLPs as second feature set. Finally, the GT features are augmented to MFCCs and PLPs. Table 7.11 summarizes the dependency of the bottle-neck size and the input size of the multi-layer perceptron.

The experiments show that the size of the bottle-neck is important when all three feature streams are combined. The best performance is obtained when the 873 input features are encoded in 75 components. When only one or two feature streams are augmented for the multi-layer perceptron training, the gain by increasing the bottle-neck is not significant. Nevertheless, a size of 50 or 75 seems to be the best choice for all feature sets and corpora.

7.5.3 Summary

In this section we analyzed the combination of several short-term features using multi-layer perceptrons with the bottle-neck structure. The bottle-neck feature combination method benefited from the feature combination by multi-layer perceptrons as well as from the bottle-neck

structure introduced in the network.

Even though we observed reasonable results by the bottle-neck feature combination method, the size of the bottle-neck played an important role. When the bottle-neck size was too small to encode the full input vector, the final recognition results got worse. Therefore, when the size of the input features was large, the bottle-neck was enlarged as well. By default, the bottle-neck should not be smaller than $\frac{1}{20}$ of the input size. The minimal bottle-neck size will be the number of phonemes of the language or 50.

7.6 Recurrent Neural Network Feature Combination

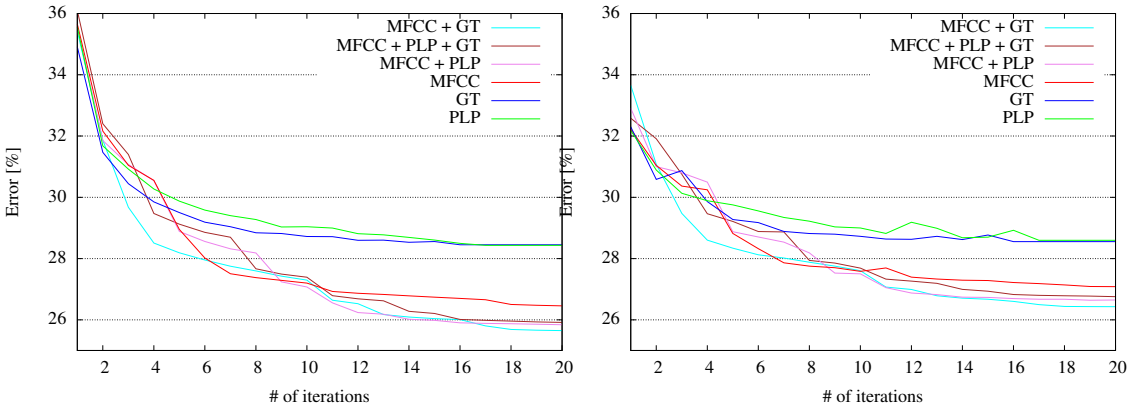
In the previous sections acoustic feature combination is performed using multi-layer perceptrons. In Chapter 5 we show that RNNs outperform the standard multi-layer perceptron based approach. Especially the long-short-term-memory topology [Hochreiter & Schmidhuber 97] combined with the bi-directional structure [Schuster & Paliwal 97] show significant improvements over the multi-layer perceptron approach. In the bi-directional approach two RNNs are trained for which the training sequence is provided in forward and backward directed order. Since each RNN contains the information up to the current time frame, the whole training sequence is provided during training. The main advantage of the long-short-term-memory structure results from the gating units which are able to cope with the problem of the vanishing gradient [Bengio & Simard$^+$ 94].

We further develop the ANN feature combination method by transferring the concept of combining several feature streams from multi-layer perceptrons to RNNs. We combine multiple feature streams, taking the long-short-term-memory topology into account. Since the bi-directional long-short-term-memory RNNs outperform all other ANN structures in this thesis, bi-directional long-short-term-memory RNNs are trained on the different augmented short-term features.

7.6.1 Experimental Results

The short-term features are preprocessed as described in Section 7.2, where we introduced the ANN based feature combination approach using multi-layer perceptrons for the first time. Each feature stream is augmented by Δ and $\Delta\Delta_1$. Depending on the number of feature streams combined, the output size varies from 33 up to 97. Due to the recurrent connection of the bi-directional long-short-term-memory RNNs and the training of a forward and backward directed long-short-term-memory RNN, temporal context is not required.

The topology of the bi-directional long-short-term-memory is kept as simple as possible. A simple bi-directional long-short-term-memory with two-layers is trained. Each long-short-term-memory contains a hidden layer of size 200. In order to keep the training time feasible, the phoneme classes are taken as output targets. On the Spanish task the bi-directional long-short-term-memory RNNs contains about 400k parameters when trained on a single feature stream and up to 500k parameters when all feature streams are taken into account. Figure 7.4 presents the progress of the frame errors of the bi-directional long-short-term-memory RNNs on the training and validation set.



(a) Training set **(b)** Validation set

Figure 7.4 Progress of the frame error during the training of the bi-directional long-short-term-memory RNNs. The error is measured on the training and validation set. The bi-directional long-short-term-memory RNNs are trained on different short-term features and their combinations.

As mentioned in Chapter 5, the frame error drops by adjusting the learning rate. Here, the learning rate is adapted whenever the frame error on the validation set is under a specific threshold or starts to increase. We start with a learning rate of $\eta = 0.0001$ and decrease it by a factor of 2 each time the relative improvement on the validation set drops under 0.2. Even though the frame error drops continuously, no difference in the final frame error of the bi-directional long-short-term-memory RNNs trained is observed. Nevertheless, the best frame error is achieved by the long-short-term-memory combining the MFCC and GT features. Figure 7.4 illustrates the process of the frame errors over the iterations of the different RNNs trained.

After the training of the bi-directional long-short-term-memory RNNs the 33 log posterior estimates are augmented by the MFCC features to train a tandem system. The general setup of the tandem system stays unchanged. A linear discriminant analysis reduces the posterior estimates within a sliding of size 9 to 45 components. Finally, these features are augmented to the 45 dimensional linear discriminant analysis reduced MFCCs. As the system trained on the multi-layer perceptron based posterior estimates, the tandem systems are trained on a 90-dimensional feature vector. Detailed information on the acoustic model for Spanish are given in Section A.3 and in the previous sections.

Table 7.12 summarizes the final feature combination results using the bi-directional long-short-term-memory RNN structure. Again, the word error rate decreases when multiple feature streams are included in the RNN training. As in the single multi-layer perceptron based feature combination method, the bi-directional long-short-term-memory RNNs benefit from the contrastive feature extraction methods. Nevertheless, the improvements from the additional feature streams become less when more feature streams are combined. Our recognition results improve slightly when all three feature streams are augmented. In contrast to the multi-layer perceptron based feature combination results, the third feature stream does not provide any significant information to discriminate the final targets further. Nevertheless, combining all

Table 7.12 Effect of bi-directional long-short-term-memory RNN feature combinations on Quaero Spanish. The tandem systems are based on MFCCs augmented by the bi-directional long-short-term-memory RNN posteriors and are speaker adapted using SAT/CMLLR. Each feature stream is reduced by linear discriminant analysis to 45 components, including a temporal context of ± 4 frames.

	BLSTM-RNN input		Testing corpora (WER [%])			
Feature type	Type	Size	dev10	eval10	eval09	dev09
MFCC	—	—	21.6	18.2	16.7	29.8
+ BLSTM-RNN	GT	31	19.9	16.6	15.2	26.4
	PLP	33	20.0	16.2	15.2	26.6
	MFCC		19.4	15.9	14.9	26.3
	+ GT	64	19.0	15.4	14.3	25.7
	+ PLP	66	18.9	15.7	14.5	26.0
	+ GT	97	18.8	15.4	14.2	25.6

features is much easier than finding which combination of two feature streams works best.

Compared to the multi-layer perceptron based feature experiments in Section 7.2, here, the absolute improvements are similar but resulting in slightly better relative improvements. Overall, the bi-directional long-short-term-memory RNN benefits from the complementary information of different short-term features and achieves the best single ANN based results.

7.6.2 Summary

In this section we successfully transferred the multi-layer perceptron feature combination technique to the RNN task. Similarly to the multi-layer perceptron based combination results we obtained large improvements by combining different short-term features. The combination of the MFCC and GT features or the combination of all three feature streams by a single bi-directional long-short-term-memory RNN achieved the best recognition performance. The quality of the final bi-directional long-short-term-memory RNN based features was influenced by the number of features combined and by the type of features combined. Instead of searching for the best combination, the network itself sorts the features by relevance and ignores the unimportant information. Due to this, the combination of all features will be recommended and the small overhead in training will be acceptable.

7.7 Stacking of Recurrent and Non-recurrent Neural Networks

As shown in the previous sections, combining several features by ANNs is very effective and efficient. In Section 7.4 we improve the combination approach by introducing a hierarchical framework. Even though we achieve good results with the hierarchical framework, a single network trained on the same feature sets has not been outperformed.

In this section we develop the hierarchical framework further by staking recurrent and non-recurrent ANNs. We will show that using posteriors derived from an RNN improves the multi-layer perceptron posterior estimates, whereas in the other way around the RNNs do not benefit from the multi-layer perceptron posteriors. Since the RNN provides good features for the multi-

layer perceptron training, the RNN should be used as a preprocessing step. Nevertheless, the performance of the RNNs posterior estimates could not be outperformed.

7.7.1 Hierarchical Processing of MLPs and RNNs

Motivated by the improvements of the hierarchical framework based on multi-layer perceptrons, we set up the training of bi-directional long-short-term-memory RNNs using probabilistic features of a previously trained multi-layer perceptron. Since the features derived from the multi-layer perceptron achieve very good results, these features represent the raw features and good target class discriminations as well.

The first experiments are performed on the Spanish es-medium corpus. As we will show in the experimental section, using the multi-layer perceptron based posterior estimates as input for the bi-directional long-short-term-memory RNNs training is not successful. Therefore, we exchange the multi-layer perceptron based posteriors estimates by the bottle-neck concept and train a bi-directional long-short-term-memory RNN on the es-small corpus only.

7.7.1.1 Input Feature: Posterior Estimates

We start the combination of recurrent and non-recurrent networks on the Spanish es-medium corpus. The log posterior estimates used as input to train the bi-directional long-short-term-memory RNNs are taken from the multi-layer perceptron described in Section 7.2. In the following experiments we choose the same configuration of the bi-directional long-short-term-memory RNNs as described in Section 7.6. The bi-directional long-short-term-memory RNNs consist of one hidden layer with 200 nodes. The normalized input features correspond to the 33 dimensional log posterior estimates of the previously trained multi-layer perceptrons.

The training accuracies of the bi-directional long-short-term-memory RNNs training show a non-promising result. After training, the final accuracies on the training and validation set are about 2% absolute worse compared to the bi-directional long-short-term-memory RNNs trained directly on the short-term features. Remember, the frame accuracies during the ANN training are not directly interpretable to measure the quality of the final features.

Table 7.13 summarizes the corresponding tandem recognition results of this hierarchical processing. The results show the same tendency as we observe on the frame accuracies during the training. Rather than to benefit from the class information encoded in the multi-layer perceptron posterior estimates, the information confuses the long-short-term-memory. This result is independent of the quality of the bi-directional long-short-term-memory RNNs features and the number of feature streams encoded in the ANN posterior estimates. Overall, the performance of the bi-directional long-short-term-memory RNNs trained on MFCCs could not be reached in any experiment. Moreover, after training the long-short-term-memories on the multi-layer perceptron posteriors, we do not achieve the same or even a similar performance compared to the tandem system. This is why the stacking of multi-layer perceptrons and RNNs is not a suitable way to combine these two ANN structures.

Table 7.13 Effect of multi-layer perceptron based posteriors for bi-directional long-short-term-memory RNN training on Quaero Spanish. The multi-layer perceptrons are trained on different short-term features and provide the input for the bi-directional long-short-term-memory RNNs, marked by ←. The tandem systems are trained on MFCCs and different ANN posteriors which are speaker adapted using SAT/CMLLR. The MFCCs and MLP-posteriors are reduced by linear discriminant analysis to 45, including a temporal context of ±4 frames and the RNN posteriors by linear discriminant analysis to 20 components without any context.

	BLSTM-RNN	MLP	Testing corpora (WER [%])			
Feature type	Input type	Type	dev10	eval10	eval09	dev09
MFCC	—	—	21.6	18.2	16.7	29.8
+ MLP-posterior	—	MFCC	20.4	16.9	15.5	28.4
+ BLSTM-RNN	MFCC		19.4	15.9	14.9	26.3
	MLP-posterior	GT	21.7	18.1	16.6	28.9
		PLP	21.8	17.9	16.5	29.2
		MFCC	21.2	17.3	16.3	29.1
		+ GT	20.6	16.8	15.7	27.4
		+ PLP	21.4	17.6	16.3	28.6
		+ GT	20.8	17.3	16.0	28.4

7.7.1.2 Input Feature: Bottle-neck

To analyze the worse performance of the bi-directional long-short-term-memory RNN posterior estimates, we perform a second experiment where we exchange the multi-layer perceptron posteriors by the bottle-neck features. The bottle-neck features do not represent the posteriors directly. They are an abstract representation of the raw input features and produce better recognition results.

We extract the bottle-neck features from a 4-layer multi-layer perceptron with three hidden layers. The first and third hidden layer contains a large number of units, whereas the second hidden layer consists of few nodes only. In our configuration, the bottle-neck is of size 50 and the other layers are set to 2000 and 1500. In the forwarding step the bottle-neck features are derived by taking the linear activation of the bottle-neck layer. Further connections after the bottle-neck are skipped.

The number of units in the hidden layer of the bi-directional long-short-term-memory RNN is reduced by a factor of 2 to 100 to decrease the number of trainings and the overall training time. This limits the number of parameters of the bi-directional long-short-term-memory RNN to 140k. As in the previous experiments, a linear discriminant analysis reduces the final posterior estimates of the bi-directional long-short-term-memory RNN down to a 20 dimensional subspace. All ANN based features are trained on MFCCs only and use the Spanish es-small corpus with 60h.

Table 7.14 summarizes the hierarchical ANN experiments on the 60h. The trainings based on the bottle-neck features show the same results as the posterior estimates in the previous section. Moreover, the number of contextual frames used in the multi-layer perceptron training has a large impact on the bi-directional long-short-term-memory RNN results which is not the case when the bottle-neck features are used in a tandem system. Even though we do not analyze the

Table 7.14 Effect of MLP-BN features for bi-directional long-short-term-memory RNN training on Quaero Spanish es-small corpus. The multi-layer perceptrons are trained on MFCCs with different context length and provide the input for the bi-directional long-short-term-memory RNNs. The tandem systems are trained on MFCCs and different ANN posteriors which are speaker adapted using SAT/CMLLR. The MFCCs are reduced by linear discriminant analysis to 45, including a temporal context of ± 4 frames and the ANN features by linear discriminant analysis to 20 components without any context.

	BLSTM-RNN	MLP-BN input		Testing corpora (WER [%])			
Feature type	Input type	Type	Context	dev10	eval10	eval09	dev09
MFCC	—	—	—	22.3	18.5	17.0	30.6
+ MLP-BN	—	MFCC	$\pm$ 4	20.7	17.1	16.1	28.6
	—		$\pm$15	20.5	17.1	15.4	28.1
	—		$\pm$25	20.6	17.2	15.5	27.7
+ BLSTM-RNN	MFCC	—		19.9	16.8	15.3	27.3
	MLP-BN	MFCC	$\pm$ 4	24.0	19.8	17.9	32.1
			$\pm$15	21.1	17.5	16.3	29.2
			$\pm$25	21.4	17.6	16.4	29.2
	+ MFCC			20.3	16.7	15.3	27.4

size of the context used by the recurrent connections in the long-short-term-memory, more than 9 frames are taken into account. By providing the baseline MFCC in addition to the bottle-neck features we get rid of the worse performance. The bi-directional long-short-term-memory RNN trained on both, the bottle-neck features and the MFCCs, achieves similar performance as the long-short-term-memory without the bottle-neck features. Thus, the effect of the bottle-neck or posterior features is questionable. Moreover, the training is more efficient and computational resources are saved when the bi-directional long-short-term-memory RNN is trained directly on the raw features.

7.7.2 Stacking of RNNs and MLPs

In the previous section we show that multi-layer perceptron based posterior estimates or MLP-BN features are not suitable as input to train an RNN. In this section we change the order of the ANN processing starting with a bi-directional long-short-term-memory RNN and use the posterior estimates as input to train a multi-layer perceptron.

7.7.2.1 Small Scale Experiments

First we test this hierarchical concept on the Spanish es-small corpus. The bi-directional long-short-term-memory RNNs are used as input to train an multi-layer perceptron and the final multi-layer perceptron posteriors are transformed by linear discriminant analysis to 20 components. Table 7.15 show the corresponding recognition results after speaker adaptation. The training of the multi-layer perceptron benefits from the bi-directional long-short-term-memory RNN based posterior estimates. The discrimination of the phoneme classes is transferred to the training of the multi-layer perceptron and improves the performance compared to the multi-

Table 7.15 Effect of bi-directional long-short-term-memory RNN based posteriors for multi-layer perceptron training on the Quaero Spanish es-small corpus. The bi-directional long-short-term-memory RNN is trained on MFCCs and provides the input for the multi-layer perceptron. The tandem systems are trained on MFCCs augmented by different ANN posteriors which are speaker adapted using SAT/CMLLR. A linear discriminant analysis reduces the MFCCs to 45 components, including a temporal context of ± 4 frames and another linear discriminant analysis projects the ANN posteriors down to 20 components without any context.

	ANN input			Testing corpora (WER [%])			
	Type	Size	Context	dev10	eval10	eval09	dev09
MFCC	—	—	—	22.3	18.5	17.0	30.6
+ BLSTM-RNN	MFCC	33	± 0	19.9	16.8	15.3	27.3
+ MLP	MFCC	297	± 4	20.7	17.1	16.1	28.6
	BLSTM-RNN			19.9	16.7	15.5	27.3

layer perceptron posteriors trained directly on the MFCCs. Even though the bi-directional long-short-term-memory RNN based tandem system could not be outperformed, the multi-layer perceptron based tandem system achieves the same performance on all corpora. In the next section we show how this is scaled to larger corpora in addition to the feature combination results. Overall, the long-short-term-memory-RNN provides good features, which can be used for multi-layer perceptron training as well as for hybrid recognitions.

7.7.2.2 Large Scale Experiments

In the preceding section we show that the multi-layer perceptron training on posterior estimates derived from a bi-directional long-short-term-memory RNN is successful. However, reversing the network topology does not result in any improvements.

In this section we analyze the effect of hierarchical ANN feature combination stacking RNNs and multi-layer perceptrons in combination with the training on multiple feature streams. The RNNs trained are the same as described in Section 7.6. The 2-layer bi-directional long-short-term-memory RNNs contain up to 500k parameters, depending on the number of features combined. The posterior estimates within a sliding window of size 9 of the bi-directional long-short-term-memory RNNs are used as input to train a multi-layer perceptron. Each 2-layer multi-layer perceptron consists of 4000 units in the hidden layer and the number of target classes corresponds to the 33 phonetic classes of the Spanish task.

Single Feature Stream

The hierarchical combination results are split into two main parts. In the first experiments we analyze the influence of the raw features of the multi-layer perceptron when the short-term features are presented as additional input. Table 7.16 summarizes these results. The multi-layer perceptron based posterior estimates within a sliding window of size 9 are transformed by linear discriminant analysis to 45 components and are augmented by the linear discriminant analysis transformed MFCCs. A tandem system is trained on top of the augmented feature vector. As in the small scale experiments, the hierarchical posterior estimates achieve the same per-

Table 7.16 Effect of bi-directional long-short-term-memory RNN based posteriors for multi-layer perceptron training on Quaero Spanish. The bi-directional long-short-term-memory RNN is trained on MFCCs and provides the input for the multi-layer perceptron. The tandem systems are trained on MFCCs augmented by different ANN posteriors which are speaker adapted using SAT/CMLLR. The features are reduced by linear discriminant analysis to 45, including a temporal context of ± 4 frames.

	ANN input			Testing corpora (WER [%])			
	Type	Size	Context	dev10	eval10	eval09	dev09
MFCC	—	—	—	21.6	18.2	16.7	29.8
+ BLSTM-RNN	MFCC	33	± 0	19.4	15.9	14.9	26.3
+ MLP	MFCC	297	± 4	20.4	16.9	15.5	28.4
	BLSTM-RNN			19.4	16.0	14.9	26.7
	+ MFCC	594		19.2	15.8	14.9	26.3

formance as the bi-directional long-short-term-memory RNN based posteriors. An additional small improvement is obtained by providing the same short-term feature during the multi-layer perceptron training. Even though this improvement is not large, the gain is up to 0.2% absolute in word error rate. We recommend using the additional features for all further hierarchical long-short-term-memory-multi-layer perceptron stacking experiments.

Multiple Feature Streams

In the second part we investigate the hierarchical long-short-term-memory-multi-layer perceptron stacking and the combination of different feature streams. The setup of the long-short-term-memories as well as the setup of the multi-layer perceptrons remains unchanged. Since the best hierarchical stacking result is obtained by providing the same features in each stage of the hierarchy, all bi-directional long-short-term-memory RNNs are augmented with their input features. Table 7.17 shows the tandem recognition results after speaker adaptation. Compared to the single feature combination results presented in Table 7.2 on 95 the RNN based posteriors improve the performance. Nevertheless, the best results of the RNN based tandem system could not be really outperformed. The hierarchical posteriors achieve the same or a slightly better performance on almost all corpora.

The frame accuracies obtained on the training and validation sets during the multi-layer perceptron training exceeds 80% correctness and results in the best frame error rates on the Spanish corpus. Therefore, the bi-directional long-short-term-memory RNN provides a good feature extraction which can be used in other ANN topologies and structures as well. Nevertheless, almost the same performance is achieved using the first network in the hierarchy (bi-directional long-short-term-memory RNN) or the second network (multi-layer perceptron).

7.7.3 Summary

In this section we investigated the hierarchical stacking of recurrent and non-recurrent ANNs. Even though the multi-layer perceptron based posterior estimates were not suitable for RNN training, it worked the other way around. The bi-directional long-short-term-memory RNN

Table 7.17 Effect of stacking bi-directional long-short-term-memory RNNs and multi-layer perceptrons for feature combinations on Quaero Spanish. The bi-directional long-short-term-memory RNNs are trained on different short-term features provided as input for the multi-layer perceptron training. The tandem systems are trained on MFCCs augmented by different ANN posteriors which are speaker adapted using SAT/CMLLR. Each feature stream is reduced by linear discriminant analysis to 45, including a temporal context of ± 4 frames.

	ANN input type			Testing corpora (WER [%])			
	MLP	RNN	Size	dev10	eval10	eval09	dev09
MFCC	—	—	—	21.6	18.2	16.7	29.8
+ MLP	MFCC	—	297	20.4	16.9	15.5	28.4
+ BLSTM-RNN		MFCC	33	19.4	15.9	14.9	26.3
+ MLP	BLSTM-RNN	MFCC	297	19.4	16.0	14.9	26.5
	+ MFCC		594	19.2	15.8	14.9	26.3
	+ GT	+ GT	873	18.9	15.4	14.4	25.6
	+ PLP	+ PLP	891	19.0	15.7	14.6	26.0
	+ GT	+ GT	1170	18.9	15.4	14.3	25.6

based posteriors provided additional information to improve the recognition performance of multi-layer perceptron based posteriors.

Providing only the RNN features in the hierarchical training led to no improvements compared to the system using the RNN features directly. To obtain improvements, the short-term features had to be presented as additional input. This behavior changed slightly when more short-term features were combined. In this case, the multi-layer perceptron achieved almost the same performance as the corresponding bi-directional long-short-term-memory RNN features.

Nevertheless, we showed that the training of the multi-layer perceptron was further improved by providing the right preprocessed features. The RNN provided a very good preprocessing of the raw input features. In addition we observed that the selection of the features presented for the RNN training is critical.

CHAPTER 8

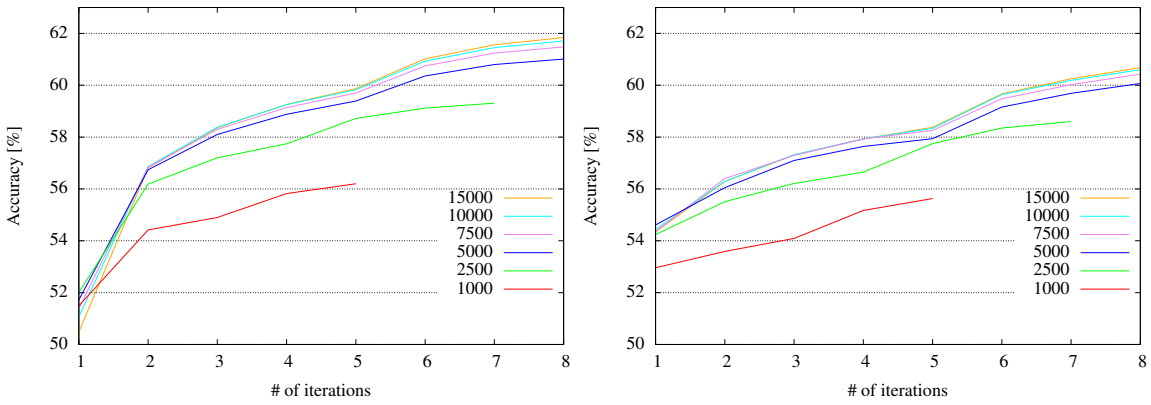
Scaling of Neural Network Parameters

In the literature no general rule exists on how to set up the configuration of a multi-layer perceptron to obtain optimal performance. The optimal number of hidden nodes as well as the number of hidden layers depends on a large number of parameters[1]:

- The number of input and output units
- The amount of training data used
- The topology/ architecture of the multi-layer perceptron
- The type of the hidden unit activation function
- The complexity of the function or classification task
- The regularization term

As a general rule, the number of nodes in the hidden layer should not be too small and not too large either. In each configuration a classification is not possible due to underfitting or overfitting to the data [Reed & Marks 99]. In the speech recognition literature the size of the hidden layers varies from some hundred nodes [Hermansky & Sharma 98] over 1,000 [Qian & Xu+ 11] and 4,000 [Plahl & Schlüter+ 10] up to 15,000 [Chen & Zhu+ 04,Stolcke & Grézl+ 06]. Moreover, in automatic speech recognition the evaluation is performed in terms of word error rate. The training of the multi-layer perceptrons as well as the training of the tandem systems does not minimize the word error rate directly. Therefore, finding the optimal parameters and configuration is not straight forward. Most of the times, the complete training pipeline including the multi-layer perceptron and the acoustic model training is needed.

[1] http://www.faqs.org/faqs/ai-faq/neural-nets/part3/section-10.html



(a) Training set **(b)** Validation set

Figure 8.1 Progress of the frame accuracy on the training (a) and validation set (b) during the multi-layer perceptron training on 230h on Chinese. The hidden layer size varies from 1,000 up to 15,000.

In this section we will investigate the correlation of the amount of training data during the training of the multi-layer perceptrons and the configuration of the multi-layer perceptrons itself. More precisely, we analyze the effect of the training data and the size of the hidden layer on a Chinese large vocabulary continuous speech recognition task. The size of the hidden layer has a big impact on the accuracy of the multi-layer perceptron. When the number of units used is too small or too large, underfitting or overfitting to the data occurs. Therefore, a trade-off between the amount of training data and the number of units has to be found. We analyze this trade-off on a Chinese task using three different size scaled corpora. To keep the network topology of the multi-layer perceptron easy, just one hidden layer will be used. The number of units in the hidden layer varies from $1,000$ up to $15,000$.

8.1 Optimizing the Hidden Layer Size

In this section we optimize the hidden layer size of a two-layer neural network w.r.t. the performance of the final tandem system. We investigate six different network sizes varying from 1,000 up to 15,000 nodes. The amount of data available on the Chinese task is 230 hours of broadcast news and broadcast conversation. Section A.1 gives detailed information on the cn-small corpus.

The six multi-layer perceptrons are trained on the 71 phonetic targets of the Chinese language including tonal information. Nine consecutive frames of the baseline MFCC features augmented by Δ and $\Delta\Delta_1$ are combined resulting in a 297 dimensional feature vector. The resulting 71 log posterior estimates of the multi-layer perceptron are combined with the MFCCs to train a tandem system. A linear discriminant analysis transforms each of the two feature streams and reduces the feature stream to a 45 dimensional vector. Figure 8.1 summarizes the performance on the training and validation set of the different hidden layer configurations during the multi-layer perceptron training.

Table 8.1 Analysis of the impact of the hidden layer size for the multi-layer perceptron training on the Chinese cn-small corpus after speaker adaptation using SAT/CMLLR. The tandem systems are trained on MFCCs and multi-layer perceptron based posteriors. Each feature stream is reduced by linear discriminant analysis to 45 components, including temporal context of size ± 4.

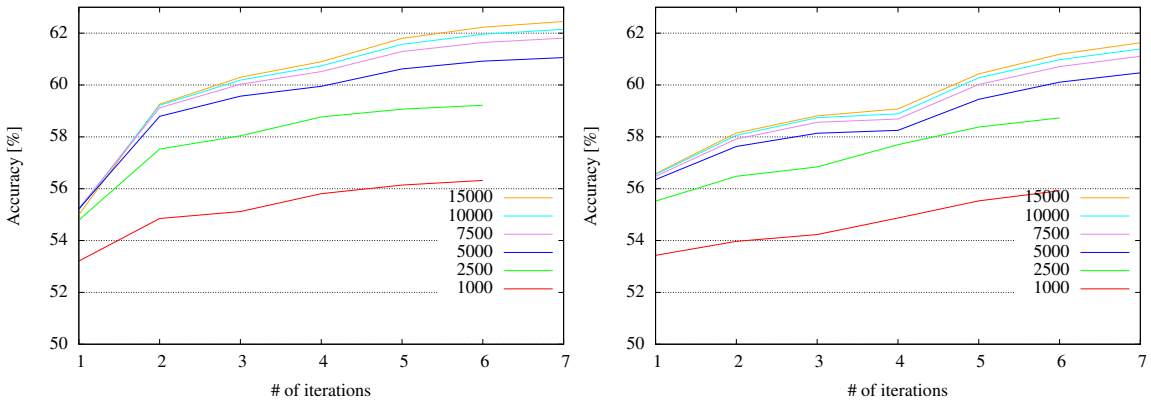
Feature type	MLP size		Testing corpora (CER [%])			
	Input	Hidden	dev07	dev08	eval08	eval07-seq
MFCC	—	—	13.8	12.9	17.4	14.7
+ MLP-posteriors	297	1,000	13.3	12.9	17.0	14.6
		2,500	13.0	12.7	16.6	14.2
		5,000	12.9	12.4	16.2	13.9
		7,500	12.7	12.4	16.3	14.0
		10,000	12.6	12.4	16.2	14.1
		15,000	12.8	12.3	16.3	14.2

Starting from 1,000 nodes in the hidden layer, the frame accuracy improves continuously when more nodes are provided. The gain becomes less when the hidden layer contains more than 5,000 nodes. The three biggest configurations achieve a final frame accuracy which differs in a range of 0.5% absolute only. Moreover, the number of multi-layer perceptron training epochs increases when the number of units in the hidden layer is enlarged. Table 8.1 lists the corresponding tandem recognition results. As suggested in Figure 8.1, we observe a similar system performance when the hidden layer contains 5,000 or more hidden nodes. When the number of nodes in the hidden layer is small (1,000 nodes), the benefit from the multi-layer perceptron features is less. Overfitting is observed when the number of units is too large (15,000). The optimal layer size for this configuration is between 5,000 and 7,500 nodes.

8.2 Scaling Network Parameters

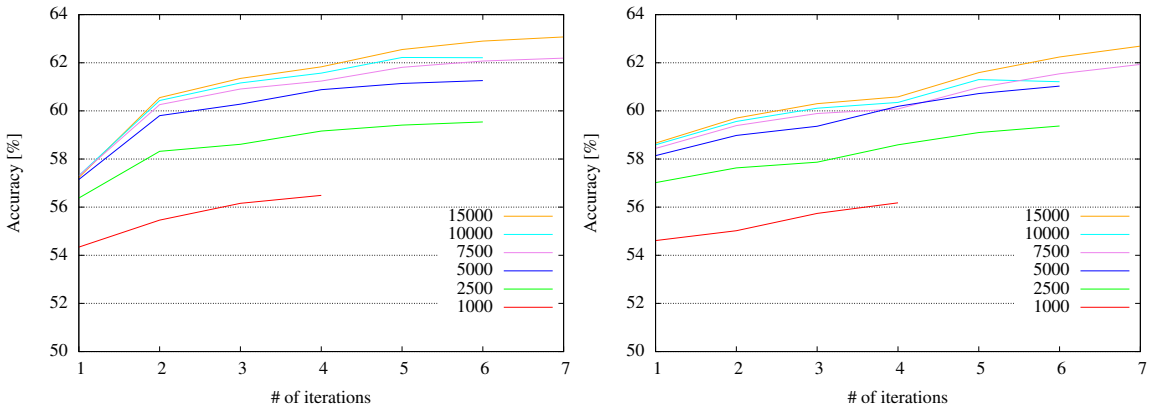
In this section we investigate the scaling of the multi-layer perceptron configuration to larger corpora. Therefore, the same configuration of the multi-layer perceptrons is tested on two larger Chinese corpora. The cn-medium is about three times larger and the cn-large corpus about seven times larger than the cn-small corpus. The general setup for the multi-layer perceptron training as well as the tandem training stays unchanged.

Figure 8.2 and Figure 8.3 show the progress in the frame accuracy on the cn-medium and cn-large corpora. As expected, the large amount of data improves the frame accuracy for each configuration. Moreover, the variance in the frame accuracy of the different configurations becomes more noticeable when additional data is provided during the training of the multi-layer perceptrons. Nevertheless, the best performance is achieved when 7,500 nodes are included in the hidden layer. The corresponding recognition results after speaker adaptation and lattice rescoring with the full language model are summarized in Table 8.2 and Table 8.3. Here, the same observation is made. Although each configuration does not behave the same on all corpora, the configuration with 7,500 nodes in the hidden layer seem to be a good trade-off between the training and decoding time of the multi-layer perceptron as well as the tandem recognition



(a) Training set **(b)** Validation set

Figure 8.2 Progress of the frame accuracy on the training (a) and validation set (b) during the multi-layer perceptron training on the cn-medium corpus. The hidden layer size varies from 1,000 up to 15,000.



(a) Training set **(b)** Validation set

Figure 8.3 Progress of the frame accuracy on the training (a) and cross validation set (b) during the multi-layer perceptron training on the cn-large corpus. The hidden layer size varies from 1,000 up to 15,000.

Table 8.2 Analysis of the impact of the hidden layer size for the multi-layer perceptron training on the Chinese cn-medium corpus after speaker adaptation using SAT/CMLLR. The tandem systems are trained on MFCCs and multi-layer perceptron-posteriors. Each feature stream is reduced by linear discriminant analysis to 45 components, including temporal context of size ± 4.

Feature type	MLP size		Testing corpora (CER [%])			
	Input	Hidden	dev07	dev08	eval08	eval07-seq
MFCC	—	—	13.8	12.6	16.8	14.0
+ MLP-posteriors	297	1,000	13.1	12.4	16.6	13.8
		2,500	12.8	12.0	16.1	13.6
		5,000	12.4	11.9	15.6	13.2
		7,500	12.2	12.8	15.7	13.1
		10,000	12.3	12.8	15.5	13.1
		15,000	12.2	12.6	15.5	13.1

Table 8.3 Analysis of the impact of the hidden layer size for the multi-layer perceptron training on the Chinese cn-large corpus after speaker adaptation using SAT/CMLLR. The tandem systems are trained on MFCCs and multi-layer perceptron-posteriors. Each feature stream is reduced by linear discriminant analysis to 45 components, including temporal context of size ± 4.

Feature type	MLP size		Testing corpora (CER [%])			
	Input	Hidden	dev07	dev08	eval08	eval07-seq
MFCC	—	—	13.7	12.6	16.6	13.7
+ MLP-posteriors	297	1,000	12.9	12.4	16.6	13.6
		2,500	12.6	11.9	15.9	13.4
		5,000	12.4	11.6	15.5	13.0
		7,500	12.3	11.5	15.3	12.8
		10,000	12.2	11.6	15.1	13.0
		15,000	12.1	11.4	15.2	12.9

performance. On the Chinese task, this result seems to be independent of the size of the corpus used. Therefore, the configuration of 7,500 nodes can be used in all multi-layer perceptron trainings on Chinese. Even though the 1,000 unit configuration achieves some improvements over the baseline, the generalization on the evaluation data is not as good as for the dev07 set used for tuning.

8.3 Summary

We investigated the scaling of the hidden layer when the amount of data was enlarged. We analyzed the training performance of the multi-layer perceptron as well as the final tandem system on three different scaled Chinese tasks.

As we showed in the experimental section, the optimal configuration of all corpora contains 7,500 nodes in the hidden layer. Whereas bigger hidden layers did not harm when the amount of data was enlarged, the improvements obtained from a bigger hidden layer was less. The op-

timal solution was similar for all multi-layer perceptron configurations. Moreover, the optimal configuration of the multi-layer perceptron was not critical. The optimal number of unit in the hidden layer could be chosen from a wide range of possible values. Each configuration differed only slightly from the other.

Independently of the amount of training data presented, the improvements obtained by small networks did not result in any large improvements. Therefore, increasing the network size was one important step to achieve reasonable improvements. Moreover, increasing the network size avoids the underfitting problem.

CHAPTER 9

Pre-training of Neural Networks

The previous chapters show that probabilistic features derived by ANNs are applied with great success to automatic speech recognition systems. Moreover, ANN based probabilistic features have become a major component of current state-of-the-art automatic speech recognition systems [Hwang & Peng$^+$ 07, Plahl & Hoffmeister$^+$ 09, Sundermeyer & Nußbaum-Thom$^+$ 11]. Unfortunately, the conventional approach to train such ANNs is limited to few hidden layers. The trained weight connections tend to get stuck in a poor local optimum when multiple hidden layers are initialized with small random values. The objective function of the ANN training is non-convex and therefore the optimal solution is not guaranteed. Recently, [Hinton & Osindero$^+$ 06] have introduced an unsupervised generative method to initialize the weight connections of deep neural networks by pre-training the weights using Restricted Boltzmann Machines.

In the following, we will analyze and compare different possibilities to pre-train the weight connections of an ANN. In general, the pre-training of an ANN can be performed in an unsupervised or supervised manner. In the supervised pre-training, the ANNs are trained by the conventional back-propagation algorithm, starting with one hidden layer and increasing the number of hidden layers one by one after each training step. In the unsupervised pre-training method, the concept of auto-encoders is taken into account. In this work we introduce two examples of the auto-encoder paradigm, the well-established Restricted Boltzmann Machines and an alternative, the Sparse Encoder Symmetric Machines. Sparse Encoder Symmetric Machines have been applied to image recognition tasks, but not yet to automatic speech recognition.

The whole section is structured as follows: First we introduce the classical initialization method using random values in Section 9.1. Section 9.2 explains the supervised initialization approach which is known as discriminative pre-training. The unsupervised initialization methods using Restricted Boltzmann Machines or Sparse Encoder Symmetric Machines are introduced in Section 9.3. Section 9.4 describes the experimental setup and the corresponding experiments. We summarize the results in Section 9.5.

9.1 Conventional Supervised ANN Training

The initialization of the weight connection of an ANN is important to speed up the training of the ANN as well as to obtain a good solution of the weight connections. Since the training of an ANN is a non-convex optimization problem, several local optima may exist. The weights of an ANN are initialized by small random values which should prevent immediate saturation and avoid symmetry in the weights [Reed & Marks 99]. Depending on the distribution of the data, [Reed & Marks 99, Table 7.1, pp. 102] suggest different weight connection initialization schemes to choose the random values.

In the following experiments as well as in all experiments already presented in this work, we have initialized the weights of all ANNs by small randomized values. The weights are randomly selected from the interval $[-0.1, 0.1]$. Furthermore, the whole ANNs are trained at once by the back-propagation algorithm. Again, the labeling of each frame of the training data is obtained from a forced alignment derived from a previously trained Gaussian hidden Markov model system.

9.1.1 Experimental Results

In this section, we briefly describe the baseline experiments using a random initialization. We focus on the concept of multi-layer perceptrons again. Nevertheless, the results obtained here are valid for other ANN topologies as well.

The experiments are performed using the hybrid recognition approach on Quaero French which is described in detail in Section A.2. The training of the multi-layer perceptrons consists of one, two or three hidden layers with 1024 hidden nodes in each layer. The alignment for the supervised multi-layer perceptron training as well as the 4501 target classes are taken from a previously trained baseline Gaussian hidden Markov model system. The target classes are the triphone states of the Gaussian hidden Markov model baseline system, clustered to 4501 states by the classification and regression tree approach. The short-term MFCC features, augmented by its first and second derivatives (Δ, $\Delta\Delta_1$), are taken as input to train the multi-layer perceptron. Depending on the number of hidden layers used, the different multi-layer perceptrons contain about 5M, or 6M, or 7M parameters respectively.

Table 9.1 summarizes the corresponding hybrid recognition results. The hybrid recognition approach is explained in detail in Section 3.1.1 of this work. The results presented here support the results in Section 3.1. Multiple hidden layers improve the performance of a multi-layer perceptron which outperforms the Gaussian hidden Markov model based system using less parameter. When the Gaussian hidden Markov model system is speaker adapted using speaker adaptive training using constrained maximum likelihood linear regression, the number of nodes in the hidden layer of a multi-layer perceptron has to be increased (see Section 3.1.1) or the number of layers. Again the number of parameters is much lower than the number of parameters of the Gaussian hidden Markov model system. We use this experimental result as baseline to evaluate the supervised and unsupervised pre-training approaches.

Table 9.1 Comparison of the hybrid recognition performance of multi-layer perceptrons with multiple hidden layers and the Gaussian hidden Markov model based systems on Quaero French. The multi-layer perceptrons are trained on triphone states clustered by classification and regression tree and short-term MFCC features. The weights of the networks are initialized by random values and each hidden layer of the multi-layer perceptron consists of 1024 nodes. The recognition systems are tuned on the development set, marked by *.

		Total # of		Testing corpora (WER [%])			
		Layers	Parameters	dev10*	eval10	eval09	dev09
GHMM	MFCC	—	50M	25.8	27.6	36.6	41.6
	+ SAT/CMLLR			24.1	25.4	33.2	38.8
Hybrid	Triphone states	2	5M	27.2	28.0	34.8	42.5
		3	6M	25.4	26.0	35.9	41.0
		4	7M	24.0	24.8	35.1	39.7

9.2 Discriminative Pre-training

In the previous section we have seen that the random initialization of the weights of a multi-layer perceptron works very well. Instead of initializing all weights randomly and train the whole multi-layer perceptron at once, the pre-training approaches train single weight connections of the network and increase the complexity of the network by and by. The main motivation for the pre-training approaches is the local optima in the loss function. Depending on the start initialization of the weights, the training can get stuck in a poor local optimum. The pre-training avoids these poor local optima by shifting the weights in the weight space to another position from where a better solution is obtained.

The simplest way to cope with the initialization problem is to perform the training of the weight connection layer by layer and to combine the single layer afterwards to construct the full neural network. This layer-wise initialization can be performed in an unsupervised [Hinton & Osindero$^+$ 06] or supervised [Bengio & Lamblin$^+$ 06] manner. After the pre-training of the weights and the construction of the network, the training of the whole network is finalized by a back-propagation step. This final step is called the *fine-tuning* step.

Training the multi-layer perceptron layer by layer leads to the concept of deep belief networks. In general, deep belief networks are probabilistic generative models. The generative models are composed of multiple layers of stochastic feature detectors. Each feature detector automatically discovers an abstract representation of the lower level features to higher level concepts [Bengio 09]. An efficient way to train such deep belief networks is described in [Hinton & Osindero$^+$ 06, Bengio 09], where each layer of the deep belief network is modeled by a Restricted Boltzmann Machine.

Instead of pre-training the weights in an unsupervised way, in this section we described the weight initialization using the discriminative pre-training method. The discriminative pre-training has been introduced by [Bengio & Lamblin$^+$ 06] and [Seide & Li$^+$ 11]. Figure 9.1 illustrates the general procedure of the discriminative pre-training. The pre-training of the weights starts with a network with just one hidden layer, an input and an output layer. After training, the output layer is removed and a new hidden layer and a new final output layer are

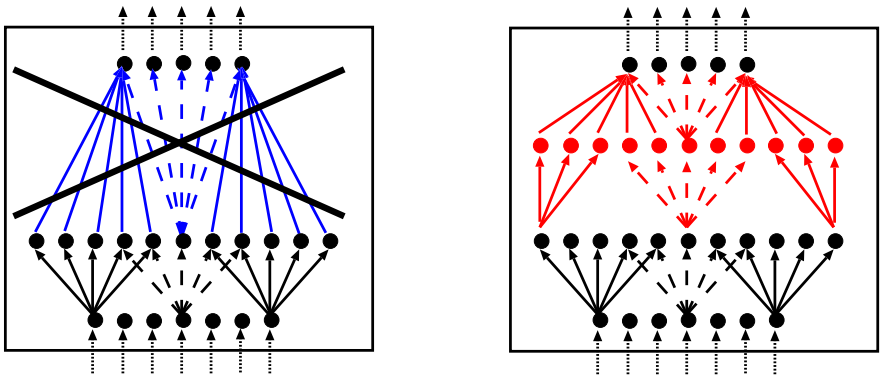


Figure 9.1 Illustration of the discriminative pre-training of the weight connections. After a supervised training of the weight connections, the output layer is removed (left figure, weight connections marked in blue). A new hidden layer is added and the corresponding weight connection to the new hidden layer and the output layer are trained (right network, marked in red). The training of the network is finalized by a fine-tuning step, training all layers of the network at once.

added. Now, the supervised training of the weight connections starts again.

Two strategies to update these new weights are known. In [Bengio & Lamblin+ 06] only the new weight connections are trained, whereas the already trained connections stay fixed. In the end, a final supervised fine-tuning step is required, where all weight connections of the network are updated during training. In [Seide & Li+ 11] the network is fine-tuned each time the number of hidden layers in the network is increased until convergence of the network is reached. In this case, a separated training step to train the new weight connections is skipped. In both approaches, the number of hidden layers is increased continuously layer by layer and the training of the weight connections is performed in a supervised fashion.

In our experiments we test both discriminative pre-training approaches. The main disadvantage of this discriminative pre-training approach is the update of the weight connections between the last hidden layer and the output layer. When a new hidden layer is added, the output layer and its weight connections are discarded.

Section 9.4 summarizes the corresponding recognition results of this discriminative pre-training method and compares this discriminative pre-training approach with other unsupervised pre-training techniques.

9.3 Unsupervised Pre-training

As described in the previous section, the pre-training of the weight connections can be performed using supervised or unsupervised training techniques. The main disadvantage of the supervised pre-training approach is that the weight connections from the last hidden layer to the output layer are always discarded, when the network is increased. In this section we investigate two different unsupervised training methods which do not have this handicap. The main motivation for the pre-training approaches is that several local optima in the loss function exist. Depending on the start initialization of the weights, the training can get stuck in one of the poor

local optima. The pre-training avoids these poor local optima by shifting the weights in the weight space to another position from where a better solution is obtained.

The most popular unsupervised pre-training method to initialize the weight connection between two layers are the Restricted Boltzmann Machines [Hinton & Osindero+ 06]. Nevertheless, the concept of Restricted Boltzmann Machines contains some disadvantages, and therefore we investigate an alternative pre-training method based on Sparse Encoder Symmetric Machines. The Sparse Encoder Symmetric Machines have been published by [Ranzato & Boureau+ 07b] for an image recognition task. We adapted the concept for automatic speech recognition.

9.3.1 Introduction and Overview

The breakthrough for the deep learning architectures has started by introducing very efficient algorithms to train such deep neural networks [Hinton & Osindero+ 06, Ranzato & Poultney+ 06, Salakhutdinov & Larochelle 10]. Each of the algorithms is based on a greedy layer-wise unsupervised pre-training approach followed by a final fine-tuning step. As mentioned in the previous section, the fine-tuning step is a back-propagation step where all weight connections are updated at once. The fine-tuning step is required to optimize the weights according to the specific target classes.

The concept of unsupervised deep neural networks or also called deep belief networks has been used first in the area of image recognition [Hinton & Osindero+ 06, Ranzato & Poultney+ 06] including a large number of further developments and analyses. [Salakhutdinov & Murray 08] performs a quantitative analysis of the deep belief network concept in general and [Salakhutdinov & Hinton 09, Salakhutdinov 09] analyze the deep network architecture and the Restricted Boltzmann Machines.

In the area of speech recognition [Mohamed & Dahl+ 09] have adapted the concept of deep belief networks and adopted the pre-training to phoneme recognition. Furthermore, the training of multi-layer perceptrons on clustered triphone states combined with the unsupervised pre-training approach using Restricted Boltzmann Machines has become a new component of current state-of-the-art automatic speech recognition systems [Mohamed & Yu+ 10, Mohamed & Sainath+ 11, Seide & Gang+ 11, Sainath & Kingsbury+ 11] and one of the most promising research areas for speech recognition of the last years.

The loss function optimized during the supervised ANN training is non convex and therefore several local optima may exist. [Bengio & Lamblin+ 06] suggest that the pre-training shifts the weight connections into a part of the parameter space where a better local optimum can be found and the optimization itself is easier. [Erhan & Courville+ 10] analyzes the question in more detail. In addition to a better generalization of pre-trained weights, the effect of adding specific constraints to the parameters during the training plays an important role. These constraints take over the role of a regularization step which is different from the normal L_1 or L_2 regularization terms [Erhan & Courville+ 10]. Furthermore, the complexity of a network with just one layer makes the training much simpler than the training of complex networks.

[Plahl & Sainath+ 12] show that the fine-tuning step applied after the pre-training does not change the global structure of the weight connections. Moreover, the fine-tuning step enlarges

Figure 9.2 Illustration of the encoder-decoder principle. The input x is encoded by the weight matrix W, resulting in the code z. The decoder reconstructs the original input starting from the code z. The performance of the systems is measured on how well the input vector is reconstructed.

the structure after pre-training to be able to discriminate the classes in the final classification task.

9.3.2 Auto-encoder

A natural way to design stackable unsupervised learning systems is based on the *encoder-decoder* paradigm [Ranzato & Boureau$^+$ 07a]. In this concept, the encoder transforms the input features x into a new representation z, which will be referred to as the *code*. Afterwards, the decoder reconstructs the input features from the code, resulting in $\hat{X}$. Figure 9.2 illustrates the encoder-decoder architecture. Typical representatives of such encoder-decoder architecture are the principal component analysis, auto-encoder neural networks, Restricted Boltzmann Machines, Sparse Encoder Symmetric Machines or de-noising auto-encoder. In this work we investigate the Restricted Boltzmann Machines and the Sparse Encoder Symmetric Machines.

In general, the unsupervised model is defined by a distribution over the input vector x, the code z and the parameters W through an energy function $E(x,z,W)$:

$$\begin{aligned} P(x|W) &= \int_{z'} P(x,z'|W) \\ &= \frac{\int_{z'} e^{-\beta \cdot E(x,z',W)}}{\int_{x,z'} e^{-\beta \cdot E(x,z',W)}} \end{aligned} \tag{9.1}$$

where β is an arbitrary constant. z' is taken from the set of all possible solutions for the codes z. The weight matrix W is updated during training to obtain the optimal code representation z. Minimizing the loss function of the encoder-decoder architecture w.r.t. the weight parameters W is equal to the negative log likelihood of the training data.

$$L(W,x) = -\frac{1}{\beta} \cdot \int_{z'} e^{-\beta \cdot E(x,z',W)} + \frac{1}{\beta} \cdot \int_{x,z'} e^{-\beta \cdot E(x,z',W)} \tag{9.2}$$

The first term is called the free energy and measures how well the input is reconstructed. The second term is the log partition function which is a penalty term. The log partition function ensures that low energy values are produced only for input pattern that have high probabilities in the (true) data distribution and high energy values for any other input pattern [Ranzato & Boureau$^+$ 07b].

The concept of the encoder-decoder architecture for unsupervised pre-training is attractive for two reasons:

1. Computing the code z after training is very fast, the encoding step or forward step requires the multiplication with W,

2. A low error value after the reconstruction (decoding step) ensures that the code captures the most relevant information.

Each pair of layers in the network can be realized by the encoder-decoder architecture. deep belief networks are constructed by stacking several of these encoder-decoders. In the next section two representatives of the concept are investigated. The Restricted Boltzmann Machines and Sparse Encoder Symmetric Machines differ mainly in the way how the log partition function is modeled.

9.3.3 Restricted Boltzmann Machines

As described in Section 1.6, the multi-layer perceptrons used in this thesis model the posterior probability $p(s|x)$ of a class or label s given the input vector x. The whole multi-layer perceptron consists of L layers, where each layer $l = 1, \cdots, L-1$ models the posterior probability $p(s^l|x^l)$ of hidden binary states s^l given the input vector x^l. The final layer L models the desired class posterior probabilities. In general, each pair of layers of the multi-layer perceptron can be realized using the encoder-decoder architecture.

The most common architecture used to pre-train the weight connections between two layers is a Restricted Boltzmann Machine. Restricted Boltzmann Machines are an effective way to initialize the weight connections of a network by unsupervised training [Hinton & Osindero$^+$ 06, Seide & Gang$^+$ 11, Sainath & Kingsbury$^+$ 11]. Each encoder-decoder distinguishes the encoder step, which consists of the forward step similar to the forward step of an ANN, and the decoder step, where the input of the encoder is reconstructed. The forward or encoding step of a Restricted Boltzmann Machine is described by:

$$\begin{aligned} f_{enc}(x^l) &= z^l(x^l) \\ &= (W^l)^T \cdot x^l + b^l_{enc}. \end{aligned} \tag{9.3}$$

The output activation is obtained after applying the activation function σ:

$$\begin{aligned} x^{l+1} &= \sigma(z^l(x^l)) \\ &= y^{l+1} \end{aligned} \tag{9.4}$$

with sigmoid activation function $\sigma(u) = \dfrac{1}{1+e^{-u}}$ (see Equation (1.17)).

The decoding step depends on the distribution of the input features. We distinguish a *Gaussian-Bernoulli* and a *Bernoulli-Bernoulli* distribution depending on the distribution for the input (visible) and the output (hidden) layer of the Restricted Boltzmann Machine. In case of a Bernoulli-Bernoulli distribution the decoding step is performed by:

$$f_{dec}(\hat{z}^l) = \sigma(W^l \cdot \hat{z}^l + b^l_{dec}), \tag{9.5}$$

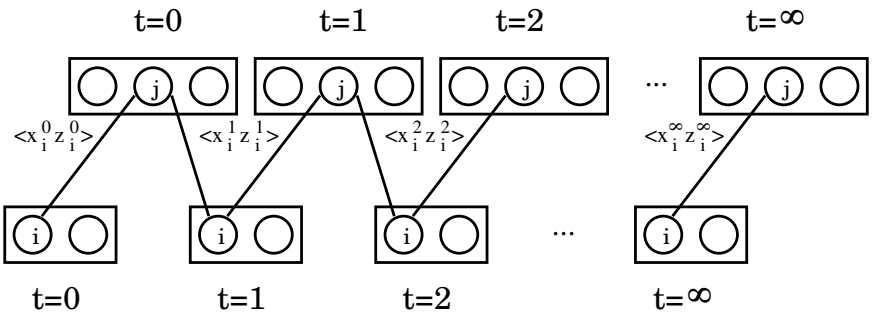


Figure 9.3 Illustration of the Gibb sampling. The encoding and decoding step is applied until convergence. In the upper row the encoded input is represented, whereas in the lower row the input and its reconstruction are shown. $\langle x_i^l z_i^l \rangle$ is the expected values after the l-th Gibb sampling step.

where $\hat{z}^l$ is a binary random sample. The decoding for the Gaussian-Bernoulli distribution is performed by

$$f_{dec}(\hat{z}^l) = W^l \cdot \hat{z}^l + b_{dec}^l. \tag{9.6}$$

The energy function of a Restricted Boltzmann Machine, including the encoder and decoder part, is described by:

$$E(x,z,W) = -z^T \cdot W^T \cdot x - b_{enc}^T \cdot z - U, \tag{9.7}$$

where $U = \frac{1}{2}(b_{dec}^T \cdot x)^2$ for the Gaussian-Bernoulli distribution and $U = b_{dec}^T \cdot x$ for the Bernoulli-Bernoulli distribution. The final loss L becomes:

$$L(x,W) = -\frac{1}{\beta}\log\sum_z e^{-\beta E(x,z)} + \frac{1}{\beta}\log\sum_{x'\in\Omega}\sum_z e^{-\beta E(x',z)} \tag{9.8}$$

where Ω is a region around the training sample. Sampling in the neighborhood of Ω is performed by an alternated Markov Chain Monte Carlo step over x and z. The Markov Chain Monte Carlo step is performed by alternating the *Gibb sampling* procedure shown in Figure 9.3. Even though the Gibb sampling has to be performed until it reaches its stationary distribution, an intermediate result is sufficient to perform the parameter updates [Hinton 10]. The update of the weight connection w_{ij} and the encoder and decoder biases are performed by:

$$\frac{\partial \log L}{\partial w_{ij}} = \langle x_i^0 z_j^0 \rangle - \langle x_i^\infty z_j^\infty \rangle \approx \langle x_i^0 z_j^0 \rangle - \langle x_i^1 z_j^1 \rangle \tag{9.9}$$

$$\frac{\partial \log L}{\partial b_{enc}} \approx z^0 - z^1 \tag{9.10}$$

$$\frac{\partial \log L}{\partial b_{dec}} \approx x^0 - x^1 \tag{9.11}$$

[Hinton & Osindero$^+$ 06,Hinton 10] give more detail on the training of Restricted Boltzmann Machines and a practical training guide. Overall, Restricted Boltzmann Machines provide an efficient method to pre-train deep belief networks or multi-layer perceptrons by approximating the contrastive divergence term using Gibb sampling.

9.3.4 Sparse Encoder Symmetric Machines

In the concept of Restricted Boltzmann Machines the log partition function is approximated by the contrastive divergence term. The Sparse Encoder Symmetric Machines do not rely on an explicit contrastive divergence term in the loss function [Ranzato & Boureau$^+$ 07b]. The log partition function is replaced by a sparseness penalty term on the output obtained by the encoder. The sparseness term allows the direct optimization of the objective function. The training of Sparse Encoder Symmetric Machines is performed by simply minimizing the average energy in combination with the additional sparseness term of the output. Similar to Restricted Boltzmann Machines, Sparse Encoder Symmetric Machines follow the encoder-decoder paradigm. The encoder and decoder are described by:

$$\begin{aligned} f_{enc}(x) &= W^T x + b_{enc} && (9.12)\\ &= z(x) \\ f_{dec}(z) &= W\sigma(z) + b_{dec} && (9.13)\\ &= \hat{x}(z) \end{aligned}$$

where the function σ is a point-wise logistic non-linearity of the form: $\sigma(u) = \frac{1}{1+e^{-\gamma x}}$ with a fixed gain $\gamma = 1$ in all our experiments.

The free energy in Equation (9.1) and Equation (9.2) of a Sparse Encoder Symmetric Machine is described by

$$E(x, z^*, W) = \alpha_e ||z^* - f_{enc}(x)||^2 + ||x - f_{dec}(z^*)||^2. \tag{9.14}$$

The free energy is divided into the difference of the current observed code z and its currently optimal solution z^*, scaled by a constant $\alpha_e = 1$, and the difference of input x and its reconstruction $\hat{x}$.

Overall, the following loss function is optimized during training, obtained from Equation (9.2) and Equation (9.14):

$$\begin{aligned} L(x, W) &= E(x, z, W) + \alpha_s \cdot h(z) + \alpha_r |W|_1 \\ &= \alpha_e ||z^* - f_{enc}(x)||_2^2 + ||x - f_{dec} z^*)||_2^2 \\ &\quad + \alpha_s \cdot h(\hat{z}) + \alpha_r |W|_1, \end{aligned} \tag{9.15}$$

where $h(z) = \sum_d \log(1 + l^2(z_d))$ and z^* is the optimal code. The loss contains the free energy (Equation (9.14)), a sparseness term ($h(z)$) as an approximation to the log partition function and a l_1-regularization term on the weights. Rather than sampling the output as for Restricted Boltzmann Machines, Sparse Encoder Symmetric Machines use the output of the encoder directly to reconstruct the input.

In order to estimate the updates for the weights and biases, the optimal code after encoding is required. Since the optimal code z^* as well as the weights and biases depend on each other, we iterate the calculation by keeping one parameter fixed. The optimal code z^* is obtained first

by optimizing $L(x,W)$ w.r.t. z by a gradient descent algorithm with fixed weights and biases. This results in the following equation to get the optimal code z^*:

$$\begin{aligned}\frac{\partial L(W)}{\partial z} &= \frac{\partial \alpha_e \cdot ||z - f_{enc}(x)||_2^2}{\partial z} + \frac{\partial ||x - f_{dec}(z)||_2^2}{\partial z} + \frac{\partial \alpha_s \cdot h(z)}{\partial z} + \frac{\partial \alpha_r |W|_1}{\partial z} \\ &= 2\alpha_e \cdot ||z - f_{enc}(x)||_2 - 2||x - f_{dec}(z)||_2 \cdot W \cdot \sigma(z)^2 \cdot e^{-z} \\ &\quad +2\alpha_s \frac{\sigma(z)^3}{(\sigma(z)-1)\cdot(1+\sigma(z)^2)}.\end{aligned} \tag{9.16}$$

The corresponding update for the weights W is calculated by:

$$\begin{aligned}\frac{\partial L(W)}{\partial W} &= \frac{\partial \alpha_e \cdot ||z - f_{enc}(x)||_2^2}{\partial W} + \frac{\partial ||x - f_{dec}(z)||_2^2}{\partial W} + \frac{\partial \alpha_s h(z)}{\partial W} + \frac{\partial \alpha_r |W|_1}{\partial W} \\ &= -2\alpha_e \cdot ||z - f_{enc}||_2 \cdot x - 2||x - f_{dec}(Z)||_2 \cdot \sigma(z) + \alpha_r \cdot f_{sign}(W),\end{aligned} \tag{9.17}$$

where $f_{sign}(u)$ returns the sign of u.

The encoder and decoder biases b_{enc} and b_{dec} are updated by

$$\begin{aligned}\frac{\partial L(W)}{\partial b_{enc}} &= \frac{\partial \alpha_e ||z - f_{enc}(x)||_2^2}{\partial b_{enc}} + \frac{\partial ||x - f_{dec}(z)||_2^2}{\partial b_{enc}} + \frac{\partial \alpha_s h(z)}{\partial b_{enc}} + \frac{\partial \alpha_r |W|_1}{\partial b_{enc}} \\ &= -2\alpha_e \cdot ||z - f_{enc}||_2 \\ \frac{\partial L(W)}{\partial b_{dec}} &= \frac{\partial \alpha_e ||z - f_{enc}(x)||_2^2}{\partial b_{dec}} + \frac{\partial ||x - f_{dec}(Z)||_2^2}{\partial b_{dec}} + \frac{\partial \alpha_s h(z)}{\partial b_{dec}} + \frac{\partial \alpha_r |W|_1}{\partial b_{dec}} \\ &= -2 \cdot ||x - f_{dec}||_2\end{aligned} \tag{9.18}$$

The complete recipe to train Sparse Encoder Symmetric Machines can be found in detail in [Plahl & Sainath+ 12]. Depending on the layer to be trained the following rules have to be kept in mind to adjust the learning rate η for the weight update (see Equation (1.40)) and the sparseness parameter α_s:

Layer-1: Choose a high value for α_s to obtain a sparse output and use a high learning rate η to achieve a lot of structure in the pre-trained weights. In our experiments we set $\alpha_s = 0.2$ and $\eta = 0.005$.

Layer-n: The output should be less sparse compared to the previous layer (current input). Decrease α_s by a factor of 2 to 4, depending on the increase/decrease of the new layer size. The learning rate η is adapted as well. Due to lower sparseness in the output, a lower learning rate is required. We decrease the learning rate by a magnitude or more.

9.4 Experimental Results

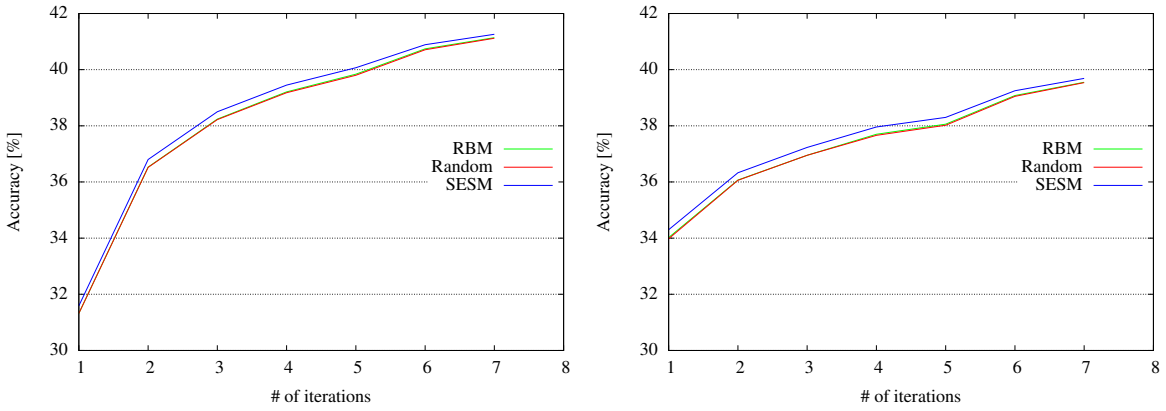
To analyze the effect of the different pre-training algorithms, we have trained several multi-layer perceptrons which differ only in the way the weights are initialized. In addition, we investigate how the effect of the pre-training behaves when the number of layers is increased. The multi-layer perceptrons trained contain one, two or three hidden layers using MFCC features, their Δ and $\Delta\Delta_1$ components and temporal contextual information of size ± 4. Overall, the input stream consists of 297 components. As shown in the experiments in Section 3.1.3.1, 4501 triphone states, clustered by the classification and regression tree approach, are provided in the output layer. During recognition, the posterior estimates of the multi-layer perceptron are used directly as state emission probabilities. As shown in Section 3.1.1, the hybrid recognition approach allows skipping the training of a tandem system. The training and the evaluations are performed on the French corpus described in Section A.2.

The network structures used in the current setup and in the experiments presented in Section 3.1.3.1 differ in the number of hidden layers and the number of nodes in the hidden layers. The multi-layer perceptrons used in the experiments consist of 1024 nodes in each hidden layer. Depending on the number of layers, the final number of parameters varies from 5M to 7M.

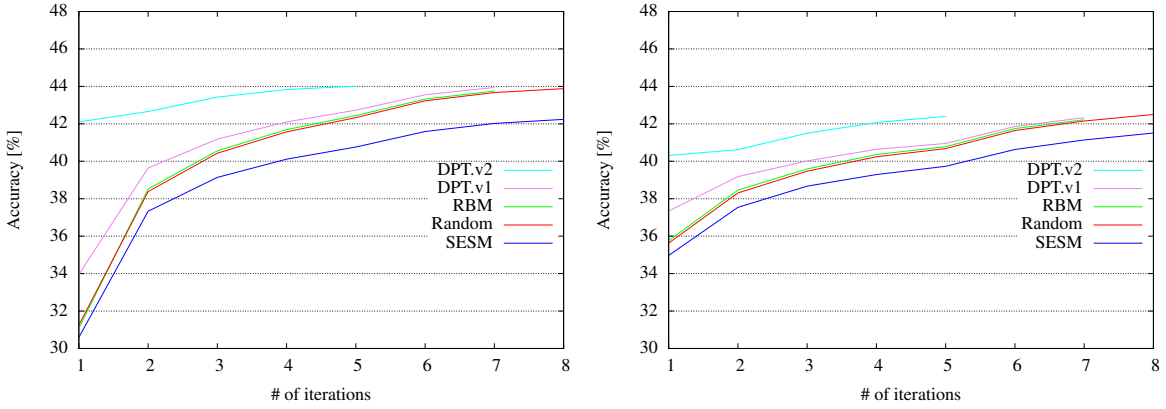
Next to the random initialization, two supervised and two unsupervised pre-training approaches are analyzed. When the pre-training of the weight connections is performed unsupervised, the weight connections are trained layer by layer using the concept of Restricted Boltzmann Machines or Sparse Encoder Symmetric Machines. Afterwards, the pre-trained weights are fine-tuned. In the supervised pre-training approach, the two possibilities differ in the way the fine-tuning step is performed. As described in Section 9.2, each layer can be pre-trained keeping the previously trained weights fixed. After the pre-training of all layers, the weights are fine-tuned. This supervised pre-training is referred to as *DPT.v2*. In the other supervised approach the training with fixed weights is skipped and only the fine-tuning step is applied. This approach is referred to as *DPT.v1*.

Figure 9.4 shows the performance of the fine-tuning steps. The different pre-training techniques achieve similar performances on the training and the validation set. The same behavior is mirrored in the corresponding hybrid recognition results of the multi-layer perceptron-posterior estimates. Table 9.2 summarizes the corresponding recognition results.

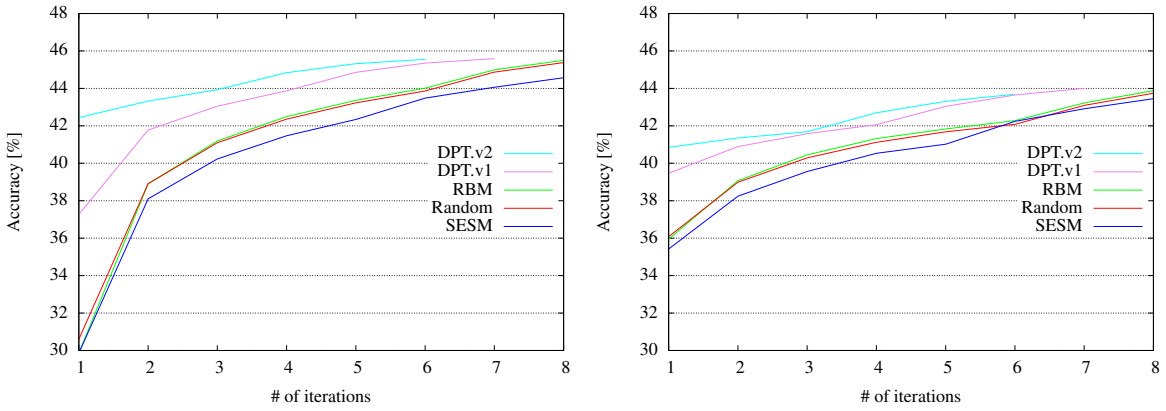
The results in Table 9.2 show a performance gap between the random initialization and the pre-trained weights. Nevertheless, the differences between the supervised and unsupervised pre-training are negligible. This result is independent of the number of layers used in the multi-layer perceptron. It is noticeable that the pre-trained weights generalize much better to unknown data than the randomly initialized weights. By increasing the number of layers the generalization effect becomes more reliable. The difference increases with each additional hidden layer by about 0.1% absolute. Overall, the Restricted Boltzmann Machine pre-trained weights achieve a 0.6% absolute better word error rate as the corresponding randomly initialized weights on the Quaero evaluation sets of 2010. The difference on the development set is only 0.2% absolute in word error rate.



(a) One hidden layer



(b) Two hidden layers



(c) Three hidden layers

Figure 9.4 Progress of the frame accuracies of the multi-layer perceptron fine-tuning step where the weights are initialized using different pre-training methods. The frame accuracies are measured on the training set (left column) and the validation set (right column) on Quaero French. The pre-training includes the concept of random values, two unsupervised methods based on Restricted Boltzmann Machines (RBMs) and Sparse Encoder Symmetric Machines (SESMs) and two supervised referred to as DPT.v1 and DPT.v2. The hidden layer size of the multi-layer perceptron varies from one (a), over two (b), to three (c).

Table 9.2 Comparison of different pre-training techniques on Quaero French. The multi-layer perceptrons differ in the number of layers trained and the initialization of the weights including random values, Restricted Boltzmann Machine (RBM), Sparse Encoder Symmetric Machine (SESM) or two supervised pre-training techniques (DPT.v1 and DPT.v2). The multi-layer perceptrons are trained on the MFCCs and the recognition is performed using the hybrid approach.

	Feature input type	Total # of layers	MLP weight Initialization	Testing corpora (WER [%]) dev10	eval10	eval09
GHMM	MFCC	—	—	25.8	27.6	36.6
	+ SAT/CMLLR			24.1	25.4	33.2
Hybrid	MFCC	2	Random	27.2	28.0	34.8
			RBM	26.9	27.6	37.3
			SESM	27.1	27.8	37.4
		3	Random	25.4	26.0	35.9
			RBM	25.1	25.7	35.8
			SESM	25.3	26.0	35.9
			DPT.v1	25.0	25.7	35.9
			DPT.v2	25.1	25.9	36.0
		4	Random	24.0	24.8	35.1
			RBM	23.8	24.2	34.7
			SESM	23.9	24.5	34.8
			DPT.v1	23.9	24.4	34.9
			DPT.v2	24.2	24.8	35.2

The best performance achieves the pre-trained weights using Restricted Boltzmann Machines. The DPT.v1 and Sparse Encoder Symmetric Machine training method result in slightly worse results. Even though the difference of the different pre-training techniques is less, the pre-trained weights achieve a better generalization than the random initialized weights. Therefore, a pre-training technique should be applied. The actual method is not important.

9.5 Summary

This section analyzed different methods to pre-train the weights of an ANN. We applied the training methods in a supervised and an unsupervised manner, increasing the network layer by layer. After the network had been pre-trained, the whole network was fine-tuned by performing the normal ANN training.

The unsupervised training was realized by the concept of the encoder-decoder paradigm. In addition to Restricted Boltzmann Machines used to pre-train the network weights, we developed a new technique based on Sparse Encoder Symmetric Machines. The main advantage of the Sparse Encoder Symmetric Machines is the direct optimization of the loss function and a clear stopping criterion reducing the number of iterations to find the optimal parameters. Using the concept of Sparse Encoder Symmetric Machines, the log partition function was modeled without any approximations and the Gibb sampling step was avoided. Nevertheless, the results

obtained by the Sparse Encoder Symmetric Machines were slightly worse compared to the Restricted Boltzmann Machine results.

In addition, we tested different fine-tuning configurations. In the first configuration we trained only the output layer until the network converges. Afterwards, we applied the back-propagation step to all layers. In the second configuration the output layer was trained for one or two epochs before the training of all layers were performed. In the last configuration we skipped the separate training of the output layer and optimized the whole network from beginning. The best performance was achieved using the third configuration followed by the first configuration. The second configuration obtained the worst results.

Overall, the pre-training of the network weights helped to obtain better recognition results. Although the improvements were small on the development set, larger and significant gains were achieved on the other testing corpora. The improvements on the testing sets increased slightly when the number of layer was increased as well. The performance differences of the pre-training methods were small. Therefore, the actual choice which method will be used to pre-train the weight was insignificant. Nevertheless, a pre-training technique should be used to obtain optimal performance. In our experiments the Restricted Boltzmann Machine approach achieved the best performance.

CHAPTER 10

Artificial Neural Networks in Image Recognition

As shown in the previous chapters, ANN based features clearly improve Gaussian hidden Markov model based automatic speech recognition systems. In automatic speech recognition the ANN based transformation of the input feature helps to discriminate the phonemes, phoneme states, triphone states or any other context dependent states. In image recognition, especially in optical character recognition and automatic sign language recognition the same statistical concepts are applied with great success [Dreuw 12].

Moreover, ANNs and deep belief networks are successfully used on different image recognition tasks. Whereas deep belief networks are used mostly to obtain a compact representation, the ANNs have been applied to extract and provide better features [Schenk & Rigoll 06, Graves & Liwicki$^+$ 09, Boquera & Bleda$^+$ 11], especially in the last years. In optical character recognition the most promising results are achieved by RNNs in combination with the long-short-term-memory structure [Graves & Liwicki$^+$ 09, Dötsch 11]. In [Gweth & Plahl$^+$ 12] the concepts of ANN based feature extraction methods are applied to automatic sign language recognition for the first time.

Motivated by those works, we transfer ANN based feature extraction with great success from speech recognition to optical character recognition [Dreuw & Dötsch$^+$ 11] and automatic sign language recognition [Gweth & Plahl$^+$ 12].

10.1 Optical Character Recognition

Similar to automatic speech recognition where the spoken utterances are translated into machine-encoded text, optical character recognition systems translate scans of handwritten text or printed text into machine-encoded text. In order to handle the optical character recognition problem, statistical methods have been proven best to deal with the large number of variations of the image data and the handwriting styles of the writer.

In the last years, ANNs have become very popular for transforming and extracting features from images as well as for classification [Graves & Liwicki+ 09, Boquera & Bleda+ 11]. Especially the RNNs in combination with the long-short-term-memory and the bi-directional network structure show improvements over the standard multi-layer perceptron based feed-forward networks [Graves & Liwicki+ 09, Dötsch 11].

This section summarizes our investigations and experiments on different ANN structures and feature preprocessing steps performed on an offline Arabic and an offline English handwriting task. Inspired by the great success of our ANN based features in automatic speech recognition, we successfully transfer the concept of ANN based features to the optical character recognition task. Including the new ANN based features in our optical character recognition system we obtain huge improvements over the baseline system.

We perform the offline optical character recognition experiments on two different corpora. Whereas the first corpus consists of isolated handwritten Tunisian town names and a closed vocabulary of less than 1,000 words, the second corpus is a large vocabulary continuous character recognition task, where the sentences are handwritten in English.

10.1.1 Isolated Word Recognition

10.1.1.1 Training and Testing Corpora

The IfN/ENIT database [Pechwitz & Maddouri+ 02, Märgner & Abed 09] (version number v2.0p1e) contains about 32,492 Arabic handwritten versions of 937 different Tunisian town names written by 1,000 writers. The 28 base characters are extended by position dependent length modeling and a separate white space model [Dreuw & Jonas+ 08]. Depending on the underlying hidden Markov model topology used in training and recognition, this results in 216 —3 hidden Markov model states per label with no repetitions— or 121 —12 hidden Markov model states per label including 2 repetitions per state— different character labels to model the Tunisian town names. The whole corpus is divided into five disjoint sets labeled from *a* to *e*. The sets *a-d* are used for training and set *e* is used for testing. In the first preliminary experiments the training is performed on sets *a-c* only. During the ANN training the sets a-d are divided further. 10% of the training material is separated to measure the performance of the ANN training on an independent validation set. Table 10.1 summarizes additional statistics of the IfN/ENIT corpus and Figure 10.1 shows typical examples of this database. Since each town name is recognized independently and no connections between these town names exist, this database belongs to the isolated word recognition tasks.

A large number of competitions are performed on this corpus. We have participated in several of these competitions and have always achieved very good recognition performances. [Pechwitz & Maddouri+ 02, Märgner & Abed 09, Märgner & Abed 10] give details about the corpus as well as on the individual competitions performed in 2009 and 2010.

Table 10.1 Corpus statistics of the IfN/ENIT corpus.

	Training and testing sets				
	Set a	Set b	Set c	Set d	Set e
# of words	6,537	6,710	6,477	6,735	6,033
# of characters	55,654	57,688	55,864	58,028	47,638
Avg. # of characters/word	8.51	8.60	8.63	8.62	7.89
Avg. image width (in pixel)	420	412	408	396	381
Avg. image height (in pixel)	98	97	94	96	93

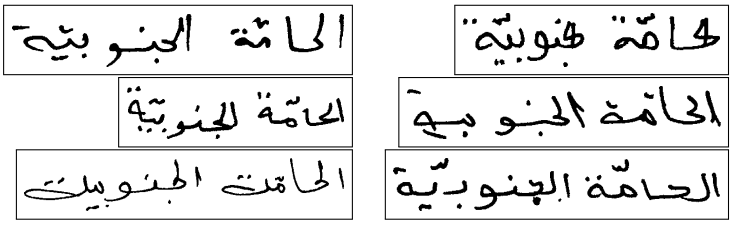


Figure 10.1 Examples of the IfN/ENIT corpus showing all the same Tunisian city town names written by different writers. We have framed the images to visualize the length of the images.

10.1.1.2 Feature Extraction

Baseline Features

Three different types of raw features are extracted directly from the image. One of the feature sets is used for preliminary tests only. We extracted these features to optimize the network structure and to find the network structure which works best. The resulting network structure has been taken to perform the experiments on two other feature sets, which achieves much better recognition performance on this corpus than the first feature set.

The first feature set consists of appearance based image slices, which are directly extracted from the raw images without any preprocessing. The slices are downscaled to a height of 16 pixels and are augmented by their temporal derivatives in horizontal direction. This 32 dimensional feature set is used to test several ANN based feature extraction methods differing in the ANN topology used.

The second feature set uses a similar feature extraction method. In contrast to the previously described first feature set, the slices are downscaled to a final size of 30 pixels only. We found that the downscaling to 30 pixels achieves better word error rates than the downscaling to 16 pixels. Afterwards, the feature vector is expanded by temporal context of size ± 4 and the 9 frames are reduced to a 35 dimensional feature subspace by principal component analysis.

The third and last feature set is the only set where the raw images are transformed in a complex manner. The images are preprocessed by the method described in [Giménez & Khoury+ 10]. A Bernoulli mixture based hidden Markov model is estimated to reposition the center of gravity of the black pixels within a sliding window. Finally, a principal component analysis reduces the center of gravity shifted features within a sliding window of size 9 to 36 components.

In the following, the three different feature sets will be referred to as *SLICE*, *SLICE-PCA* and *COG-PCA* features respectively.

Neural Network Features

In the preliminary experiments on this corpus we test a large number of different multi-layer perceptron structures. Each of the multi-layer perceptrons trained consists of one hidden layer of size 2000 and an output layer where the outputs correspond to the 216 character labels. The first multi-layer perceptrons are trained on the 16 dimensional SLICE features augmented by their first temporal derivatives.

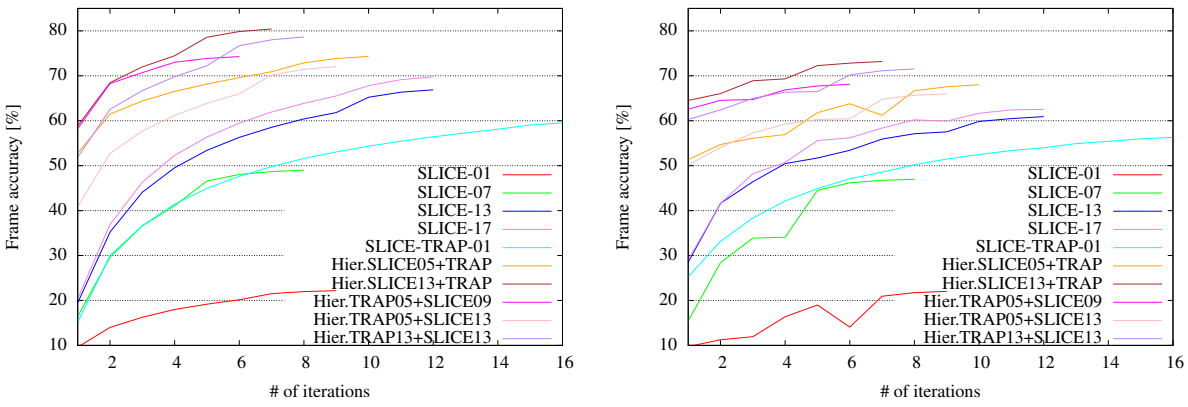
The other networks are trained on long temporal features. As described in Section 3.4.3 long-term features model long temporal dependencies in the feature set. Often the stress markers or points corresponding to special characters are shifted. This occurs often when the text is written fast. Therefore, we transform the SLICE features by the TRAP-DCT transformation, discussed in Section 3.4.3.2. The preprocessing of the SLICE features by the TRAP-DCT approach concatenates 17 consecutive frames. A discrete cosine transform reduces these 17 adjacent frames to 8 components. Overall, this results in a $32 \times 8 = 256$ dimensional feature vector. In the following this feature set will be referred to as *SLICE-TRAP*.

In the preliminary experiments we train several multi-layer perceptrons directly on the SLICE features with different context lengths and another multi-layer perceptron on the SLICE-TRAP features. In addition to these networks, we set up multi-layer perceptrons combining the previously trained multi-layer perceptrons in a hierarchical framework. Table 10.2 summarizes all configurations of the single and hierarchical multi-layer perceptrons. As mentioned above, the size of the hidden layer of the two-layer multi-layer perceptron is fixed to 2000 nodes and the outputs of the network correspond to the 216 character labels. During training the frame accuracy of the different multi-layer perceptrons is measured on the training and validation sets, summarized in Figure 10.2.

The configuration of the multi-layer perceptrons trained on the other two baseline features is slightly modified. The output of the network corresponds to the 121 character labels resulting from the length modeling using a 6-2 hidden Markov model model. The hierarchical processing in combination with the TRAP-DCT features shows an excellent training performance. The configuration of the Hier.TRAP05+SLICE09 features is transferred to the SLICE-PCA features resulting in the *Hier.TRAP05+SLICE-PCA09* feature set. As we will show in the experimental section, we observe from the preliminary results that the multi-layer perceptrons trained show a tendency of overfitting. Therefore, we reduce the number of units in the hidden layer of the first and second multi-layer perceptron in the hierarchy to 750 and 1500 nodes respectively. The log posterior estimates of the first multi-layer perceptron in the hierarchy are reduced by principal component analysis down to 95% of the variability of the eigenvalues. The 87 dimensional principal component analysis reduced multi-layer perceptron-posteriors and the SLICE-PCA features are combined by the second multi-layer perceptron using a different context length for each feature stream. The combined feature vector contains 750 components in total including temporal context of the two input features of size 5 and 9. The same hierarchical ANN configuration on the COG-PCA features are used to extract the *Hier.TRAP05+COG-PCA09*

Table 10.2 Configuration of single and hierarchical multi-layer perceptrons trained on the 216 character labels and a hidden layer size of 2000. The multi-layer perceptron based posterior estimates undergo a logarithm and a principal component analysis transformation. The principal component analysis reduces each feature set to 36 components.

	MLP input		
Feature name	Type	Context	Size
SLICE-01	SLICE	±0	32
SLICE-07	SLICE	±3	224
SLICE-13	SLICE	±6	416
SLICE-17	SLICE	±8	544
SLICE-TRAP-01	SLICE-TRAP	±0	256
Hier.SLICE05+TRAP	SLICE-13	±2	
	SLICE-TRAP	±0	436
Hier.SLICE13+TRAP	SLICE-13	±6	
	SLICE-TRAP	±0	724
Hier.TRAP05+SLICE09	SLICE-TRAP-01	±2	
	SLICE	±4	438
Hier.TRAP05+SLICE13	SLICE-TRAP-01	±2	
	SLICE	±6	566
Hier.TRAP13+SLICE13	SLICE-TRAP-01	±6	
	SLICE	±6	806



(a) Training set

(b) Validation set

Figure 10.2 Progress of the frame accuracy on the training (a) and validation set (b) during the multi-layer perceptron training on the IfN/ENIT corpus. The multi-layer perceptron configuration and the input features used to train the multi-layer perceptron based posterior estimates are described in Table 10.2. The learning rate η is adapted according to the performance on the validation set.

multi-layer perceptron based posterior estimates.

In addition to the hierarchical multi-layer perceptron based posterior estimates, we train bi-directional RNNs on the SLICE-PCA and COG-PCA features. Due to the problem of the vanishing gradient the bi-directional long-short-term-memory RNN concept is selected (see Section 5.4). As in the training of the multi-layer perceptrons, the output targets of the long-short-term-memory correspond to the 121 character labels. The bi-directional long-short-term-memory RNN contains two hidden layers of size 100 and 200. The posteriors estimates derived from the long-short-term-memory RNN are transformed by logarithm.

All ANN based posterior estimates are transformed by logarithm. Afterwards, a principal component analysis reduces all features within a window of size 9 to 64 components. Different tandem systems are trained on this 64 dimensional feature vector. Additional details to the configuration of the multi-layer perceptrons and bi-directional long-short-term-memory RNNs trained on the SLICE-PCA and COG-PCA features are given in [Dötsch 11].

10.1.1.3 Experimental Results

Preliminary Multi-layer Perceptron Results

In the preliminary experiments we train the different multi-layer perceptrons only on the training sets a, b and c. On the one hand, the posterior estimates used in the hybrid and tandem recognition approaches therefore represent the first three training sets. Moreover, all hybrid recognition results on the set d and set e are based on the posteriors produced by the same ANN configuration.

On the other hand, the training data used to train the tandem acoustic model depends on the testing set. The acoustic model is trained on set $a-c$ or on set $a-d$ and the testing is performed on set d or set e respectively. The acoustic model is trained on the principal component analysis reduced posterior estimates using a simple Gaussian hidden Markov model based model with 3 states per label without any repetitions. In total, this Gaussian hidden Markov model configuration results in 646 mixtures and about 55k densities. During the training of the multi-layer perceptron the performance of the current configuration is measured on the training and a validation set. Figure 10.2 summarizes the progress of the performances over the training iterations. We observe that a large temporal context is indispensable to achieve suitable frame accuracies and that the hierarchical structure benefits from the raw and TRAP-DCT transformed features presented at different stages of the hierarchy.

These results are mirrored in the hybrid and tandem recognition results as well. Whereas the hybrid recognition results benefit from the hierarchical structure which improves the best recognition performance by about 35% and 25% relative on set d and set e, the word error rate of the tandem systems are not significantly reduced. Nevertheless, the tandem approach achieves the best word error rates, but the difference compared to the hybrid results becomes less. The results of the hybrid and tandem recognition are summarized by Table 10.3.

The SLICE-13 features and the Hier.SLICE05+TRAP features achieve the best tandem recognition performances, whereas the best hybrid recognition performances are obtained by the Hier.SLICE05+TRAP , and the Hier.SLICE13+TRAP and the Hier.TRAP05+SLICE09

Table 10.3 Comparison of hybrid and tandem recognition performance of different multi-layer perceptron based posterior estimates on the IfN/ENIT corpus. The multi-layer perceptron based posterior estimates are transformed by logarithm and reduced by principal component analysis to 36 components. The SLICE features for the baseline Gaussian hidden Markov model system undergo a principal component analysis reduction to 30 components including a temporal context of ± 4 frames. The baseline acoustic model is improved by discriminative training using margin-based minimum phoneme error (margin-based MPE).

		Testing corpora (WER [%])			
		Hybrid		Tandem	
Feature type		set d	set e	set d	set e
SLICE		—	—	7.8	16.8
+ margin-based MPE		—	—	6.1	15.4
MLP-posteriors	SLICE-01	48.3	72.2	—	—
	SLICE-07	9.0	20.2	4.2	8.7
	SLICE-13	5.3	14.1	3.6	7.7
	SLICE-17	8.8	15.4	4.3	9.4
	SLICE-TRAP-01	7.4	18.3	4.6	9.4
	Hier.SLICE05+TRAP	3.5	10.7	3.4	7.6
	Hier.SLICE13+TRAP	3.6	10.9	3.4	10.0
	Hier.TRAP05+SLICE09	4.1	11.3	4.7	9.5
	Hier.TRAP05+SLICE13	4.9	13.4	4.4	8.9
	Hier.TRAP13+SLICE13	4.6	12.2	4.5	10.0

features. All of the last three feature sets have in common that they are trained using the hierarchical framework. The optimal configuration for the hybrid and the tandem approach has been chosen differently. Nevertheless, the feature set labeled as Hier.TRAP05+SLICE09 seems to be a good compromise between the hybrid and the tandem approach and is used in the following experiments.

In addition, we obtain that even including the margin-based maximum mutual information training criterion to improve the baseline system, the multi-layer perceptron feature based hybrid and tandem systems perform better. This verifies the result of Section 3.3, where the minimum phoneme error trained baseline systems show the same or slightly worse recognition performance than the speaker adapted tandem system including the multi-layer perceptron features.

Recurrent and Non-recurrent Networks

In the second part of the experiments we train ANN based posterior estimates using multi-layer perceptrons and RNNs based on the SLICE-PCA and COG-PCA features. Instead of using 216 character classes we reduce the number of characters to 121. As described in [Dötsch 11] we obtain a much better word error rate using the reduced character set. The final recognition results of the multi-layer perceptrons and long-short-term-memory RNN features trained on the SLICE-PCA and COG-PCA features are listed in Table 10.4. By preprocessing the input features of the ANN an additional significant improvement is obtained. Therefore, pre-processing

Table 10.4 Comparison of hybrid and tandem recognition results of the multi-layer perceptron and bi-directional long-short-term-memory RNN features trained on the SLICE-PCA and COG-PCA features on the IfN/ENIT corpus. The training of the ANNs and the tandem systems are performed on the set $a-d$ [Dötsch 11].

Feature type	ANN input Type	Set e (WER [%]) Hybrid	Tandem
SLICE-PCA	—	—	13.1
COG-PCA	—	—	6.4
MLP-posteriors	SLICE-PCA	10.3	5.9
	COG-PCA	6.6	4.7
BLSTM-RNN-posteriors	SLICE-PCA	8.7	7.2
	COG-PCA	5.8	5.0

of the input features helps to improve the overall system performance.

[Dötsch 11] shows that in general the long-short-term-memory RNN structure outperform the multi-layer perceptron on different image recognition tasks when trained on the same feature set. Similar same result has been obtained in Chapter 5 on an automatic speech recognition task and in Section 10.1.2 on another image task. This corpus is one of the few examples that the multi-layer perceptron sometimes achieves better results than the bi-directional long-short-term-memory RNNs.

10.1.2 Large Vocabulary Recognition

10.1.2.1 Training and Testing Corpora

In total, the IAM corpus [Liwicki & Bunke 05] consists of 1,539 pages with 5,685 sentences in 9,862 lines. Each English word is built of a set of 79 symbols containing lower- and uppercase letters as well as punctuation and question marks and a white space model. The database itself is divided into a training set and two sets for testing. In our setup, one of the test sets is used for parameter tuning, the other for testing only. Compared to the tuning set, the training and testing corpus contain about 6× and 3× more data respectively. Additional corpus statistics are summarized in Table 10.5. An example of the IAM database is given in Figure 10.3.

Table 10.5 Corpus statistics of the IAM database.

	Training and testing corpora		
	Training	dev	test
# of words	53,884	8,717	25,472
# of characters	219,749	33,352	100,762
# of text lines	6,192	920	2,781
Avg. # of word/line	8.75	9.47	9.16
Avg. # of characters/word	4.08	3.82	3.96
Avg. image width (in pixel)	1,751	1,740	1,763
Avg. image height (in pixel)	123	115	131

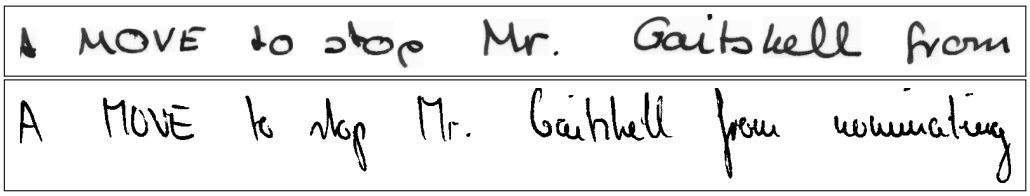


Figure 10.3 Typical training examples taken from the IAM corpus.

In recognition a 3-gram language model is applied. As proposed by [Bertolami & Bunke 08] we use the three additional text corpora Lancaster-Oslo-Bergen, Brown and Wellington to estimate our language model. The language model is smoothed by the Kneser-Ney approach [Kneser & Ney 95]. The 3-gram language model contains the 50k most common English words and has an out-of-vocabulary rate of 4.01% and 3.47% on the tuning and testing set.

10.1.2.2 Feature Extraction

Baseline Features

Similar to the baseline feature extraction method used in Section 10.1.1.2, we extracted appearance based slices directly from the raw image. In order to compensate for variations in Latin writing, we do slope and slant correction of the raw image and normalize the size of the characters. After the preprocessing of the raw images, each image is downscaled to a height of 16 pixels, while keeping their aspect ratio. The slices extracted from the image are augmented by their spatial derivatives in horizontal direction. We will refer to this feature set as *RAW-SLICE*.

[Boquera & Bleda$^+$ 11] suggest performing the preprocessing of the raw images using a cascade of different ANNs. The ANNs take over the slope and slant removal as well as the normalization step.[1] As the final step we extract the slice features from the modified images which we call *MLP-SLICE* features.

In the final baseline recognition system, we concatenate seven consecutive frames of these features to incorporate temporal and spatial context and reduce the feature to a size of 30 components using principal component analysis.

Neural Network Features

On both baseline feature sets the hierarchical multi-layer perceptron as well as the long-short-term-memory RNN are trained separately. The alignment for the ANN training corresponds to the 79 character symbols and the whitespace model and is obtained from the baseline systems. In [Dreuw & Dötsch$^+$ 11] we have shown that the training alignment for the ANN training has a significant impact. Therefore, we use the margin-based minimum phoneme error trained model to produce a forced alignment of the training corpus and train the multi-layer perceptrons. Since the alignments used in our automatic speech recognition systems are already good, the margin-based minimum phoneme error based alignment does not have such a significant impact. The

[1] A special thank goes to Salvador España Boquera from the Department of Information Systems and Computing at the Polytechnic University of Valencia for providing the multi-layer perceptron preprocessed images.

amount of training data used in automatic speech recognition as well as the speaker adapted acoustic models are the main reason why the corresponding forced alignment is really good. The retraining of the acoustic model results only in tiny modifications of the alignment.

Similar to the configuration described in Section 10.1.1.2, we set up several hierarchical ANNs. As input for the multi-layer perceptron the TRAP-DCT transformed RAW-SLICE features as well as the RAW-SLICE features themselves are used. The first multi-layer perceptron in the hierarchical ANN is trained on the RAW-SLICE augmented by its first temporal derivatives without any temporal context. The second network is trained on the 40 dimensional linear discriminant analysis reduced log posterior estimates of the first network. In addition to these multi-layer perceptron features, the same input features as in the first network are augmented and expanded by a temporal context of ± 4 frames. The hierarchical ANN consists of 1500 and 3000 nodes in the hidden layer of the first and second multi-layer perceptron. We apply the same setup to the Hier.SLICE09+SLICE09 feature which are based on the RAW-SLICE features.

In the previous section we observe that the raw features outperform the TRAP-DCT features. Nevertheless, we have trained a third hierarchical ANN which is based on the 256 dimensional TRAP-DCT transformed RAW-SLICE features. As usual, the final log posterior estimates of the first network are reduced by linear discriminant analysis from 80 to 40. The reduced posteriors within a sliding window of size ± 2 and the RAW-SLICE or MLP-SLICE features within a sliding window of size ± 4 are combined by the second network. This results in a 344 dimensional feature vector. Whereas the hidden layer in the first network contains 1500 nodes, we increased the number of nodes in the hidden layer of the second multi-layer perceptron to 3000. The total number of parameters during the multi-layer perceptron training reaches 1.6M.

The bi-directional long-short-term-memory RNNs trained on the two feature sets consist of two hidden layers where the first hidden layer has 100 nodes and the second 200 nodes. Instead of the posterior estimates of the RNN, the normalized linear output of the first layers is taken. The concept of such a bottle-neck and the performance of the bottle-neck features are explained in detail in Section 4.3. Overall, each RNN consists of less than 200k parameters.

Independently of how the ANN based features are trained, the tandem systems use principal component analysis reduced posterior or probabilistic features only. Before we apply the principal component analysis transformation, the ANN features within a sliding window of size ± 3 are combined. The bottle-neck features derived from the bi-directional long-short-term-memory RNNs are reduced from 700 to 20 components whereas the multi-layer perceptron based log posterior estimates are reduced from 560 to a final size of 30.

10.1.2.3 Experimental Results

On the IAM corpus, each character is modeled by 5 states and two repetitions resulting in a 10-state left-to-right Gaussian hidden Markov model with 391 mixtures and 25k densities and a globally pooled diagonal covariance matrix. The two baseline systems are trained on the principal component analysis reduced RAW-SLICE or MLP-SLICE features. The principal component analysis transformation of the features take into account a temporal context of ± 3 frames. As described in the previous section, the ANN based features are transformed in the

Table 10.6 Comparison of the hybrid and tandem approach on the IAM corpus using the RAW-SLICE features as input. The ANN based posterior features used in the tandem systems are reduced by principal component analysis, keeping 95% of the variability. The tandem results are improved further by the margin-based maximum mutual information (M-MMI) or margin-based minimum phoneme error (M-MPE) criterion. The hybrid recognition results are performed on the ANN based posterior estimates. In the tandem system these posteriors are reduced by principal component analysis.

Feature type		GHMM Input size	Testing corpora (WER [%]) Hybrid dev	Hybrid test	Tandem dev	Tandem test
RAW-SLICE		32	—	—	31.9	38.6
+ M-MMI			—	—	25.8	31.6
+ M-MPE			—	—	24.3	30.0
MLP-posteriors	Hier.TRAP05+SLICE09	30	31.2	36.9	26.7	33.7
	Hier.SLICE09+SLICE09		34.3	40.4	25.7	32.9
+ M-MPE			—	—	22.6	28.7
BLSTM-RNN	RAW-SLICE	20	23.4	28.1	23.0	28.4
+ M-MPE			—	—	22.2	27.0

Table 10.7 Comparison of the hybrid and tandem approach on the IAM corpus using the MLP-SLICE features as input. The tandem results are improved further by the margin-based minimum phoneme error (M-MPE) criterion. The hybrid recognition results are performed on the ANN based posterior estimates. In the tandem system these posteriors are reduced by principal component analysis.

Feature type		GHMM Input size	Testing corpora (WER [%]) Hybrid dev	Hybrid test	Tandem dev	Tandem test
MLP-SLICE		32	—	—	27.2	34.7
+ M-MPE			—	—	24.2	30.3
MLP-posteriors	Hier.SLICE09+SLICE09	32	26.5	34.2	24.6	31.0
+ M-MPE			—	—	22.9	29.0
BLSTM-RNN	MLP-SLICE	20	20.6	24.8	19.4	23.8
+ M-MPE			—	—	17.9	21.5

same way. The final feature vector contains 30 or 20 components corresponding to the type of network structure used to train the features. All acoustic models are improved retraining the acoustic model using the margin-based maximum mutual information or the margin-based minimum phoneme error criterion [Dreuw & Heigold$^+$ 09, Dreuw & Heigold$^+$ 11].

The recognition results of the different ANN features on the IAM database are summarized in Table 10.6 and Table 10.7. Although the hybrid recognition results of the TRAP-DCT based hierarchical posteriors perform better than the other hierarchical ANN features, the tandem features behave the other way around. This result supports the observation that the optimal configuration for the tandem and hybrid approach differ. Even more, the tandem approach benefits from a lower context encoded in the features and the complementary information which

Figure 10.4 Three examples taken from the SIGNUM corpus. The tracking of the dominant hand is marked by the rectangular in yellow.

is more suitable to distinguish the classes in the final Gaussian hidden Markov model based system.

As observed in Chapter 5, the bi-directional long-short-term-memory RNN approach outperforms the multi-layer perceptron based feature extraction and achieves the best recognition performance on this corpus in this work. Again, this result is independent of the input features used to train the ANNs.

10.2 Sign Language Recognition

Sign language is the most natural communication for deaf people. The information in sign language is presented visually by hand, torso, and facial expressions. The main challenge in the area of automatic sign language recognition is to extract and combine all these sources in order to obtain the best recognition performance. Sign language itself is not international. Therefore, a large number of different sign languages exist. All of these sign languages are developed independently of each other such as the American sign language, the French sign language or the German sign language. Moreover, each sign language is affected by regional influences resulting in a large number of regional dialects.

In this thesis we use automatic approaches to track the dominant hand of the signer from the image or video directly and to extract the features in order to train a Gaussian hidden Markov model based system. This allows recording the gestures by camera and avoids the need of gloves or other auxiliary means.

After the extraction of the appearance based features, different multi-layer perceptrons transform these features further. As shown in [Gweth & Plahl+ 12] the multi-layer perceptron based feature extraction approach used in this thesis outperforms the current state-of-the-art features and improves the recognition system.

10.2.1 The SIGNUM Corpus

The SIGNUM database contains German sign language. As described in [von Agris & Kraiss 07], the SIGNUM database is recorded under laboratory conditions with a uniform background and dark clothes for the signer. The whole corpus contains different speakers out of which we select just one for our experiments. Figure 10.4 shows three examples of the SIGNUM corpus

Table 10.8 Statistics of the speaker dependent SIGNUM corpus.

	Training	Testing
# sentences	1809	531
# frames	417k	114k
# running glosses	11k	2.8k
OOV rate [%]	—	0.6

and the speaker selected. Table 10.8 gives additional statistics of the SIGNUM corpus. The vocabulary of the corpus contains the 455 most frequent signs used in everyday conversation. In recognition, a 3-gram language model is applied with a perplexity of 97.5 on the test set.

10.2.2 Feature Extraction

10.2.2.1 Appearance based Features

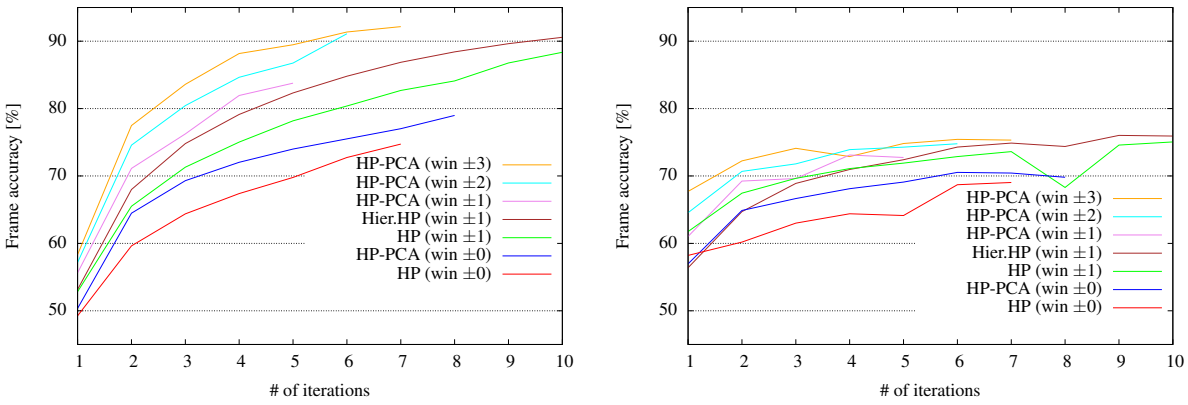
We extract the appearance based features directly from the image. The main advantage of this appearance based feature extraction methods is that these methods do not rely on any external model or information of additional sensors.

The appearance based features used in this thesis for the automatic sign language recognition task are similar to the features presented in [Zahedi & Keysers$^+$ 05a, Zahedi & Keysers$^+$ 05b]. We first track the dominant hand of the signer using a modified version of the algorithm presented in [Dreuw & Deselaers$^+$ 06]. The tracking algorithm optimizes the position of the hand over the whole sequence and avoids local decisions. Afterwards, we crop the dominant hand from the image, centered at the center of the hand with a total size of 32×32 pixels. These feature set is referred to as *hand patch (HP)*.

10.2.2.2 Neural Network based Features

In the previous chapters we have analyzed a huge number of ANN based feature extraction methods. Since no ANN based features have been used so far in the area of automatic sign language recognition, a simple model is used as startup.

The ANN feature extraction is based on a simple 2-layer multi-layer perceptron as shown in Figure 4.1 (a). The hidden layer consists of 2500 nodes and the output targets correspond to the 455 glosses of the SIGNUM corpus and an additional silence model. The network is trained on two different appearance based feature sets. The first feature set consists of the hand patches derived from the tracking of the dominant hand with a total size of $32 \times 32 = 1024$ Pixel. Since these hand patch features are huge, a principal component analysis reduces the hand patches to a lower dimensional feature space. As for other features, temporal derivatives are necessary to obtain the best performance. Therefore, all features within a sliding window of size ± 2 (5×1024) are reduced by principal component analysis to 200 components. The full hand patch features are referred to as *HP* and the reduced hand patch features as *HP-PCA*.



(a) Training set

(b) Cross validation set

Figure 10.5 Frame accuracies on the training and the validation set during the multi-layer perceptron training. The input of the network is expanded by using adjacent frames. The learning rate η is adjusted according to the performance on the validation set.

In the training of the multi-layer perceptrons different contextual information is included as well. Due to the large size of the hand patch features, a maximal context size of ± 1 is used, whereas up to ± 3 frames of the HP-PCA features are included in the input. In the forwarding step the 456 dimensional posterior estimates are transformed by logarithm and reduced by principal component analysis to 200 components including a temporal context of ± 2 frames. This is the same setup as we use for the baseline acoustic model allowing a simple comparison between the baseline features and the multi-layer perceptron based features.

We augment the posterior estimates of the currently best multi-layer perceptron by the hand patch features into a new 2-layer multi-layer perceptron. This second network in this hierarchy is trained on the combined feature stream including a temporal context of size ± 1. Overall, this results in an input vector of size 4440. To provide enough model power the hidden layer size of the multi-layer perceptron is increased to 4000 units.

Figure 10.5 shows the frame accuracies during the training of different multi-layer perceptrons. As expected, the performance increases when more temporal context is included in the ANN training. It is surprising that the training accuracy exceeds 90% but the cross validation still stays under 75%. Despite a robust estimation of the parameters, the multi-layer perceptron seems to memorize the training data and therefore the multi-layer perceptron will provide poor posterior estimates and a poor generalization to unknown data. In order to overcome this problem, we train the hierarchical multi-layer perceptron, but the training and validation performance show a similar tendency.

Table 10.9 Comparison of multi-layer perceptron and non multi-layer perceptron based features on SIGNUM. The tandem systems are trained on the full image, the hand patches or the multi-layer perceptron based posteriors. All feature streams are reduced by principal component analysis to 200 components, including a temporal context of ± 2 frames.

	MLP input feature			
Feature type	Type	Context	Size	Testing (WER [%])
Image (full)	—	—	—	27.2
HP (hand patch)	—	—	—	16.0
MLP-posteriors	HP	± 0	1024	14.6
		± 1	3072	13.9
	HP-PCA	± 0	200	13.8
		± 1	600	13.0
		± 2	1000	14.7
		± 3	1400	17.4
	Hier.HP-PCA +HP	± 1	4440	15.7

10.2.3 Experimental Results

The experiments are performed on the SIGNUM corpus with the same speaker in the training and test set. The acoustic model is trained on three feature sets, the full image, the hand patches and the multi-layer perceptron based posterior estimates. A principal component analysis transforms each feature stream to a final size of 200 to keep the size of the input feature set the same. Taking the same size of the input features after principal component analysis allows comparing the quality of the features directly. We start the training of the acoustic model from a linear segmentation and update the alignment after each Gaussian hidden Markov model training iterations. Each gloss is modeled by six states with two repetitions resulting in a 3-2 left-to-right Gaussian hidden Markov model model with a total number of 1,365 mixtures and 30k mixture densities. The training of each Gaussian hidden Markov model based model is performed independently of each other.

As expected, the experimental results in Table 10.9 show that the multi-layer perceptron based posterior estimates significantly outperform all other features. This result is independent of the features used as input to train the multi-layer perceptron. In contrast to automatic speech recognition, the size of the context frames used for training the multi-layer perceptron plays an important role here. Even though the training performance of the multi-layer perceptron is increased by a larger context, the size of the context is critical. When the context used in the multi-layer perceptron training is too large, the performance of the posterior estimate based Gaussian hidden Markov model tandem system increases dramatically. The main reason is that the multi-layer perceptron includes no regularization term in the training function and tends to memorize the data instead resulting in a poor generalization of the posterior estimates as shown in Table 10.9. Nevertheless, when the configuration of the multi-layer perceptron is correct, the final features provide the best information to recognize the 456 different glosses.

Table 10.10 Comparison of the effect of different feature combination methods on SIGNUM. The multi-layer perceptron-posteriors and hand patch features are combined using feature concatenation, log-linear model combination or system combination.

	Testing (WER [%])
HP (hand patch)	16.0
MLP-posteriors	13.0
Combination level:	
Feature (concatenation)	13.1
Model (log-linear)	12.8
System (ROVER)	12.5

Note that we trained the hierarchical multi-layer perceptron to overcome the overfitting problem and the poor generalization of the posterior estimates. As shown in the Table 10.9, the hierarchical multi-layer perceptron features do not achieve any better recognition results. Even more, the performance drops significantly when the hierarchical ANNs are trained. The huge number of parameters in the network and the small amount of training data result in poor posterior estimates.

10.2.4 Combination Results

Independently of the best multi-layer perceptron configuration, the overall performance can be increased further by combining different systems. There are currently three approaches to combine the systems. In Section 3.2 we achieve the best recognition performance combining the baseline and the multi-layer perceptron features on different automatic speech recognition tasks, and system combination achieves similar performance only. Next to feature and system combination approaches the different acoustic models can be combined using a log-linear model combination framework [Zolnay & Schlüter$^+$ 05].

Table 10.10 summarizes the comparison results of the three combination methods used. We combine the best baseline system using the hand patches and the best multi-layer perceptron based tandem system. Even though several system combination approaches have been applied ROVER [Fiscus 97] obtains the best recognition results.

In contrast to what we observe in the experiments in automatic speech recognition, we do not improve the acoustic model when the acoustic model is trained on the multi-layer perceptron based posterior estimates and the baseline hand patch features. Moreover, combining the principal component analysis reduced hand patches and the principal component analysis reduced multi-layer perceptron features do not even achieve any improvements. This correlates with the huge feature vector size (400) and the large number of parameters in the Gaussian hidden Markov model system. The best combination approach for two systems is obtained by system combination. This changes when more feature streams are combined. By providing the full image as a third component, the log-linear model combination approach and system combination approach perform equally well. Both combination approaches achieve a final word error rate of 11.9% [Gweth & Plahl$^+$ 12]. Thus, the feature concatenation approach is outperformed by any other combination approach on this automatic sign language recognition task.

10.3 Summary

We successfully transferred the concept of ANN based features on two offline optical character recognition tasks and an automatic sign language recognition task. In each of these tasks the hybrid integration of the ANN based posterior features was outperformed by the tandem approach. Although the posterior estimates were improved significantly by the hierarchical framework and the gap between the hybrid and tandem recognition results became smaller, the tandem approach achieved the best recognition performance. Independently of the way how the ANN probabilistic features were integrated, the baseline Gaussian hidden Markov model systems were outperformed by the hybrid approach and by the tandem approach.

Due to the amount of training data available in the optical character recognition and automatic sign language recognition tasks, multi-layer perceptrons tended to overfit to the data when the number of temporal contextual frames was increased. In both tasks a temporal context, which was chosen to be too large, dramatically reduced the quality of the multi-layer perceptron based features resulting in poor posterior estimates. In the hierarchical framework the size of the context did not have such a critical impact. Nevertheless, the number of temporal frames used in the multi-layer perceptron training was optimized w.r.t. the final word error rate to reduce the negative influence.

In optical character recognition we found that the temporal pattern based feature extraction is outperformed by just using the raw features. This verified the results obtained for the automatic speech recognition task in Section 3.4.3.4. Given this, we skipped testing long-term features in the automatic sign language recognition task. In addition, we found that the bi-directional RNNs with the long-short-term-memory structure achieved the best recognition results.

To our best knowledge, the ANN based features were applied for the first time on an automatic sign language recognition task. We showed that multi-layer perceptron based posterior estimates significantly outperform all other classical appearance based features and achieved the best single system performance. The extraction of the appearance based features did not rely on any additional knowledge sources, e.g. the marker of gloves of the signer. The data driven appearance based feature extraction and the ability of ANNs to extract the important information from the data lead to the great success of multi-layer perceptron based features. In addition, the ANNs had the possibility to discriminate a huge number of classes with fewer parameters compared to the Gaussian hidden Markov model based systems. Furthermore, the final system was improved by combining several models in the log linear framework or by using system combination. Both, system combination and a log-linear model combination lead to similar performance and achieved the best published recognition performance on the SIGNUM corpus. In contrast to the experiments for automatic speech recognition, the combination of multi-layer perceptron based features and the baseline features used to train the multi-layer perceptron did not improve the system performance on the optical character recognition and the automatic sign language recognition tasks.

CHAPTER 11

Scientific Contributions

The aim of this thesis was to investigate ANN based features for large vocabulary continuous speech recognition systems. The same concepts were applied to optical character recognition and automatic sign language recognition systems. The integration of the ANN probabilistic features focused on the tandem approach and the ANN training on phonemes as target classes.

In this chapter the different scientific goals that have been defined in Chapter 2 will be revisited and it will be analyzed in how far these goals had been accomplished:

- Chapter 3 analyzed the hybrid and the tandem integration approach of ANN based features into hidden Markov model based automatic speech recognition systems. The hybrid approach achieved competitive or even better results than the Gaussian hidden Markov model baseline system, when the ANN training was performed on triphone states or context dependent states. The tandem approach achieved competitive results compared to the hybrid approach when both systems were trained on the same complex feature sets. Nevertheless, the main advantage of the tandem approach was that the Gaussian hidden Markov model based adaptation techniques like speaker adaptive training or discriminative training could be applied without any additional effort.

- Furthermore, in Chapter 3 the tandem feature integration approach was optimized and the effect of speaker adaptive training and discriminative training was analyzed. Augmenting the multi-layer perceptron features to the MFCCs achieved better performance than when using the multi-layer perceptron features only. As in the baseline system, speaker adaptive training and discriminative training improved the system performance further. The best performance was achieved after discriminative training. Moreover, a discriminatively trained baseline system did not outperform the multi-layer perceptron based tandem system after speaker adaptation.

- Additionally, the effect of different feature pre-processing steps for the ANN training was investigated. The effect of speaker adaptation or any other linear transformed applied to the features was negligible after the acoustic model had been adapted to compensate speaker variances. Significant differences were only recognized in the speaker independent case. Providing long-temporal contextual information up to 600ms did not result in any improvements either.

- Chapter 4 and Chapter 5 investigated different ANN topologies. The multi-layer perceptron topologies covered in this work were the bottle-neck processing and the hierarchical framework. A new topology was developed combining the hierarchical framework and the bottle-neck structure. The new ANN structure benefited from each approach and combined the advantages of both topologies. Especially the concept of long-term multi-resolution RASTA benefited from the hierarchical MLP-BN topology. Therefore, the splitting of the multi-resolution RASTA features into its fast and slow modulation frequencies was necessary.

  The concept of RNN did not outperform the multi-layer perceptron in general. Due to the vanishing gradient problem the long-short-term-memory structure was required. The best RNN results were obtained by the bi-directional long-short-term-memory structure, which outperformed the multi-layer perceptron features using much less parameters. The long-short-term-memory concept was applied the first time to a large vocabulary continuous speech recognition task.

- Chapter 6 and Chapter 8 investigated the effect of the training of cross-lingual ANN features and the scaling of the network parameters, when the amount of training data was increased. Cross-lingual features, where the training of the network parameters was performed on another language as the decoding, generalized very well to other languages. Moreover, the cross-lingual features outperformed the intra-lingual features. The excellent performance of the cross-lingual features was mostly derived from the large amount of training data and the hierarchical bottle-neck structure. That's why the kinship of the language played only a minor role.

  Therefore, training just a single ANN for multiple languages simplified the system development circle and saved computational resources.

- In Chapter 7 several multi-layer perceptron based feature combinations to simplify the development circle of the multi-layer perceptron features have been investigated. Since the previously feature combination techniques like feature concatenation or linear discriminant analysis had may limitations, the most promising approach to combine different acoustic features was system combination. Different subsystems had to be trained on each acoustic feature set and the hypotheses of the different subsystems were combined afterwards. The developed multi-layer perceptron based feature combination technique did not have the same limitation as other feature combination approaches. The main advantage of using ANNs for feature combination was the non-linear transformations of the features.

The new combination method outperformed the system combination approach providing the complementary information of the different feature sets at an early stage.

- Additionally, the behavior of the different ANN topologies during the ANN based feature combination has been analyzed. The hierarchical feature combination benefited from the different features provided, but did not outperform the combination approach using a single network.

  The bottle-neck size played a significant role during the feature combination. When the bottle-neck was too small, the advantage of the bottle-neck vanished.

- In Chapter 9 the weight initialization of a deep neural network had been analyzed. The pre-training was performed in a supervised and an unsupervised way. Further, a new unsupervised training method called Sparse Encoder Symmetric Machine was developed. The Sparse Encoder Symmetric Machines got rid of the limitations of the Restricted Boltzmann Machine approach and the Sparse Encoder Symmetric Machine pre-training method performed as good as the Restricted Boltzmann Machine method. Moreover, Sparse Encoder Symmetric Machines had a better stopping criterion than Restricted Boltzmann Machines and the number of training needed is less.

  The actual difference in the pre-training methods was less. Therefore, the pre-training method in practice did not play any significant role. Nevertheless, the pre-trained weights generalize much better to unknown data than the randomly initialized weights.

- Chapter 10 showed that the ANN based features were not limited to automatic speech recognition. The ANN based features had been successfully transferred to optical character recognition and automatic sign language recognition and achieved mostly the same results as for automatic speech recognition. In contrast to the results for speech, the tandem system trained on the two image tasks did not benefit from the combination of the ANN features and the raw feature stream. The best results were obtained when just the ANN based feature stream has been used. In automatic speech recognition the short-term features outperformed the long-term features. In the two image tasks the behavior was the same.

APPENDIX A

Corpora and Systems

In this work, experiments are conducted on the three languages Chinese (Mandarin), French and Spanish. Whereas most experiments are performed on the Spanish task, the French and Chinese corpora are taken into account to answer specific questions and to verify the results obtained on the Spanish task.

Section A.1 introduces the Gale Chinese corpora and corresponding systems followed by the description of the Quaero French system in Section A.2. Finally, Section A.3 describes the Quaero Spanish corpora and systems in detail.

A.1 Gale Chinese System

The transcription of Chinese broadcast news and broadcast conversation has been one of the sub-tasks of the global autonomous language exploitation project. Within this project different Chinese systems have been developed [Hoffmeister & Plahl$^+$ 07,Plahl & Hoffmeister$^+$ 08,Plahl & Hoffmeister$^+$ 09] based on short-term features and features derived from ANNs.

A.1.1 Corpora

The training and testing corpora, summarized in Table A.1, consist of the Hub4 corpus and speech data collected within the global autonomous language exploitation project (releases: Y1Q1-4, P2R1-2, P3R1-2, and P4R1). Whereas the 30h of the Hub4 corpus are carefully transcribed, the global autonomous language exploitation data uses quick transcription.

The *cn-small* corpus is equally distributed over broadcast news and broadcast conversation, whereas the *cn-medium* corpus is built from the data releases Y1Q1-4. In addition to the quick transcription, the cn-small and cn-medium corpora consist of 30h of data of the Hub4 corpus. The *cn-large* corpus is created by using all quick transcriptions of the releases Y1Q1-4 and

Table A.1 Corpus statistics for Gale Chinese.

Corpus				
Type	Name	#Segments	#Words	Audio data [h]
Training	cn-small	206K	2.4M	230
	cn-medium	658K	7.0M	700
	cn-large	1.3M	16.2M	1,580
Testing	cn-dev07	1,655	27.5K	2.5
	cn-eval07-seq	1,013	28.1K	1.6
	cn-dev08	618	10.5K	1.0
	cn-eval08	1,888	49.1K	2.8

P2R1-2 and P3R1-2 and P4R1 of the global autonomous language exploitation project but excludes the Hub4 data. The different training corpora are used e.g. to analyze the scaling of the network parameters during the ANN training.

The development and evaluation data are also provided within the global autonomous language exploitation project. Each of the testing corpora contains about 2h of speech data on average. During decoding, the parameters are tuned on the development corpus cn-dev07, whereas all other corpora are used for testing.

A.1.2 Neural Network Training

The trainings of the ANNs are performed on all three training corpora. Since the training of an ANN is supervised, a Gaussian hidden Markov model based system is trained beforehand to provide the training labels [Plahl & Hoffmeister+ 09]. The target labels of the ANN training are derived from a forced alignment created by the previous trained Gaussian hidden Markov model. This alignment is also used for the training of the acoustic model afterwards.

The ANNs are trained on the Chinese corpora using the multi-layer perceptron architecture only. The networks are trained by the QuickNet Tool[1] developed at the International Computer Science Institute.

The trained ANNs are based on these following components:

Input Features

- Short-term features: MFCC, PLP, GT
- Long-term features: temporal pattern, multi-resolution RASTA
- Temporal context up to ± 4 frames (not included in the features)
- Mean and variance normalization

Network Topology

[1] http://www.icsi.berkeley.edu/Speech/qn.html

- Multi-layer perceptron
- Simple network: one hidden layer
- Hidden layer size: 7,500 (tested: 1,000 − 15,000)
- Hierarchical and bottle-neck structure

Output layer

- Target classes: 71 tonemes
- Softmax normalization

Others

- Performance measuring on training and validation set
- Learning rate adjustment according to performance on the validation set
- Early stopping to avoid over-fitting

The main question to be answered on the Chinese tasks concerns the scaling of the network parameters used in the hidden layer (see Chapter 8). Therefore, several multi-layer perceptrons are trained with 1,000 and 2,500 and 5,000 and 7,500 and 10,000 and 15,000 nodes in the hidden layer for each of the three training copora.

To combine several short-term features the training of additional multi-layer perceptrons are performed on the cn-small corpus. The multi-layer perceptrons trained combine one, two or three feature streams (see Section 3.4.1). According to the best configuration found in Chapter 8, the size of the hidden layers for these experiments is fixed to 7,500 nodes.

The same configuration is used for the long-term feature experiments on the cn-small corpus. Different multi-layer perceptrons are trained on the TRAP-DCT and the multi-resolution RASTA features. The hierarchical network topology is applied during the training because of the splitting of the multi-resolution RASTA features into fast and slow modulation frequencies (see Section 3.4.3).

Overall, a huge number of different multi-layer perceptron based posterior estimates are provided for acoustic training. The training of the multi-layer perceptrons is performed on 93% of the corresponding corpus. The other 7% of the corpus is used as validation set to measure the performance of the training and to adjust the learning rate. The random selection into training and validation set is made once for each of the three training corpora, but kept fixed for all multi-layer perceptron trainings.

A.1.3 Acoustic Modeling

For each of the different multi-layer perceptron based posterior feature streams, a speaker independent and a speaker adapted acoustic model is trained using the tandem approach. The systems are speaker adapted using SAT/CMLLR. Some acoustic models are further improved by discriminative training using the margin-based minimum phoneme error criterion. In addition to a baseline system for each training corpus, we train an initial system as described

in [Plahl & Hoffmeister+ 09]. The alignment derived from this acoustic model is used to provide the training alignment for the multi-layer perceptron trainings as well as for the acoustic model training. Overall, several different systems are trained independently of each other on the Gale Chinese task.

Similar to the training setup described in [Plahl & Hoffmeister+ 09] the acoustic models consist of:

Input features

- Short-term features (16 dimensional)
- Tonal feature (1 dimensional)
- Posterior estimates derived from a multi-layer perceptron (71 dimensional)
- Feature reduction to 45 components by linear discriminant analysis including ± 4 frames
  - Short-term and tonal features $((16+1)\times 9 = 153)$
  - Posterior estimates derived by a multi-layer perceptron $(71 \times 9 = 639)$
- Total Gaussian hidden Markov model input feature size: $45+45 = 90$

acoustic model

- 3×1-state hidden Markov models
- Cross-word acoustic model
- State-tying via phonetic decision tree (CART)
- 4,501 mixtures with a total of 1.1M Gaussian densities

The acoustic training of all individual subsystems is performed by the RASR toolkit developed at our department at the RWTH Aachen University [Rybach & Gollan+ 09].

A.1.4 Decoding

The whole decoding process is divided into three decoding passes. As in [Plahl & Hoffmeister+ 09] the decoding setup used is described by:

- 1st decoding pass: maximum likelihood trained vocal tract length normalization adapted acoustic model (fast variant of vocal tract length normalization)
- 2nd decoding pass: maximum likelihood trained speaker adapted acoustic model, using speaker adaptive training using constrained maximum likelihood linear regression and maximum likelihood linear regression
- 3rd decoding pass: lattice re-scoring with full 4-gram language model

Table A.2 Statistics of training and testing corpora for Quaero French.

Corpus		#Segments	#Words	Audio data [h]
Type	Name			
Training	fr-train	160K	2.7M	216
Testing	fr-dev09	2,755	68.8K	5.9
	fr-eval09	1,356	41.0K	3.7
	fr-dev10	2,478	37.0K	3.8
	fr-eval10	1,866	31.7K	2.7

The word list contains 60k words and the language model is a large 4-gram with a perplexity on the development of $PP_{dev07} = 367$. A pruned version of the language model is used during decoding and the decoding with the full 4-gram is applied as the last recognition step using lattice re-scoring. Word lists and language model are kindly provided by Speech Technology and Research Laboratory/University of Washington and are equivalent to the models used in the Speech Technology and Research Laboratory/University of Washington global autonomous language exploitation evaluation systems [Hwang & Peng+ 07, Lei & Wu+ 09].

A.2 French Quaero System

The main goal of the Quaero project is to analyze, classify and extract information from different sources like speech, and image, and video, and text for several European languages. Within this project the RWTH developed speech recognizers for several languages [Nußbaum-Thom & Wiesler+ 10, Sundermeyer & Nußbaum-Thom+ 11]. In the following the Quaero French system used in this work is described in detail.

A.2.1 Corpora

The audio data contains broadcast news and broadcast conversation as well as podcasts downloaded from the world wide web. Table A.2 shows the statistics of the training and testing corpora. In addition to the data provided within the Quaero project, the French training corpus consists of the ESTER and the ESTER2 corpus [Sundermeyer & Nußbaum-Thom+ 11]. Overall, 216h of speech data is available for training the acoustic model and the ANNs.

The development and evaluation corpora are provided within the Quaero project. Each of the testing corpora contains about 3h or more of speech data. During decoding, the parameters are tuned on the development corpus of 2010 (dev10), whereas all other corpora are used for testing.

A.2.2 Neural Network Training

The general training procedure for the multi-layer perceptron trainings for French follows the general training procedure on the Gale Chinese (see Section A.1.2). Again, a previously trained Gaussian hidden Markov model system provides the labels for the training of the multi-layer

perceptron. The multi-layer perceptrons are trained on phoneme classes as well as triphone states. The triphone states are clustered according to the state tying used in the baseline Gaussian hidden Markov model. This state tying is obtained by a classification and regression tree which is created by asking specific phonetic questions [Beulen 99]. [Plahl & Schlüter$^+$ 11a,Sundermeyer & Nußbaum-Thom$^+$ 11] provide additional information about the Quaero French system. The described system has been used to provide the labels for the multi-layer perceptron training as well as the initial alignment for the acoustic training.

On the French corpora only different multi-layer perceptron architectures have been used. The multi-layer perceptrons are trained by the QuickNet Toolkit[2] developed at International Computer Science Institute.

The general configuration of the multi-layer perceptrons trained on Quaero French is summarized by:

Input Features

- Short-term features: MFCC, PLP, GT
- Temporal context up to ± 4 frames
- Mean and variance normalization

Network Topology

- Multi-layer perceptron
- Simple network: one hidden layer
- Hidden layer size: 4500 nodes
- Optional: pre-training of the weights
  - Up to three hidden layers
  - Each hidden layer contains 1024 nodes

Output layer

- 3 different target classes
  - Phoneme classes: 44
  - Phoneme states: 130
  - Triphone states (CART): 4501
- Softmax normalization

Others

- Performance measured on a training and a validation set
- Learning rate adjustment according to performance on the validation set
- Early stopping to avoid over-fitting

[2] http://www.icsi.berkeley.edu/Speech/qn.html

- Initialization of the weights using supervised or unsupervised pre-training

The French training corpus is chosen to perform all hybrid recognitions. Therefore, several complete multi-layer perceptron trainings are performed with and without pre-training of the weight connections. Chapter 9 describes the pre-training of the weight connections in more detail. For comparison, phoneme and phoneme state posteriors as well as triphone state posteriors are trained using the same MFCC based feature stream as input. The MFCC features contain the first 16 components, their first derivatives (Δ) and the second derivative of the first component ($\Delta\Delta_1$).

Overall, several multi-layer perceptron based posterior estimates are provided for acoustic training. The training of the network is performed on 93% of the whole corpus. The other 7% of the corpus is used to measure the performance of the training and to adjust the learning rate. The random selection into training and cross-correlation set is made once and kept for all multi-layer perceptron trainings.

A.2.3 Acoustic Modeling

For each multi-layer perceptron trained on phonemes a speaker independent and a speaker adapted acoustic model is set up using the tandem approach. The system is speaker adapted by speaker adaptive training using constrained maximum likelihood linear regression. Hybrid recognitions are performed for all multi-layer perceptrons trained on clustered triphone states.

In addition to the initial baseline system, a new baseline system is trained using MFCC features only. This baseline system is described in detail in [Plahl & Schlüter$^+$ 11a]. The alignment derived from this initial model is used to provide the training alignment for the multi-layer perceptron training as well as for the acoustic model training.

The configuration of the acoustic model can be summarized by the following list:

Input features

- Short-term features (16 dimensional)
- Posterior estimates derived from a multi-layer perceptron (44 and 130 dimensional)
- Feature reduction by linear discriminant analysis of 9 adjacent input frames to 45 components
  - Short-term MFCCs ($16 \times 9 = 144$)
  - Posterior estimates derived from an multi-layer perceptron ($44 \times 9 = 396$; $130 \times 9 = 1170$)
- Total Gaussian hidden Markov model input feature size $45 + 45 = 90$

Acoustic Model

- $3-1$-state hidden Markov model
- Cross-word acoustic model
- State tying via phonetic decision tree (CART)
- 4,501 mixtures with a total of 1.1M Gaussian densities

A.2.4 Decoding

The whole decoding process is divided into two main passes. As in [Plahl & Schlüter+ 11a, Sundermeyer & Nußbaum-Thom+ 11] the decoding setup used is described by:

- 1st decoding pass: maximum likelihood trained acoustic model using vocal tract length normalization adapted features (fast variant of vocal tract length normalization)
- 2nd decoding pass: maximum likelihood trained speaker adapted acoustic model using speaker adaptive training using constrained maximum likelihood linear regression and maximum likelihood linear regression

During decoding a word list of 200k words and a 4-gram language model is used. The language model is trained using the Speech Technology and Research Laboratory language model Toolkit [Stolcke 02], smoothed by the modified Kneser-Ney method. [Sundermeyer & Nußbaum-Thom+ 11] gives more details on the language model used. The language model perplexities on the different corpora are summarized in Table A.3.

Table A.3 Additional corpus statistics for Quaero French.

	Testing corpora			
Corpus	fr-dev10	fr-eval10	fr-eval09	fr-dev09
perplexity	171.3	215.5	197.3	201.5

A.3 Spanish Quaero System

As mentioned in the previous section the main goal of the Quaero project is to analyze, classify and extract information from different sources like speech, image, video and text for several European languages. Starting from the Spanish system in [Lööf & Gollan+ 07] we further develop the speech recognizer for Quaero Spanish. The Spanish system presented here is based on the system described in [Plahl & Schlüter+ 11b], which uses the es-small corpus.

A.3.1 Corpora

As for Quaero French, the audio data contains broadcast news and broadcast conversation as well as podcasts downloaded from the world wide web. All audio data used to train the different ANNs and the acoustic models for Spanish are provided within the Quaero project. Table A.4 summarizes the statistics of the training and testing corpora for Quaero Spanish. During decoding, the parameters are tuned on the development corpus of 2010 (dev10), whereas all other corpora are used for testing.

Table A.4 Corpora statistics for the Spanish Quaero systems.

Corpus				
Type	Name	#Segments	#Words	Audio data [h]
Training	es-small	20K	0.7M	59
	es-medium	40K	2.0M	158
Testing	es-dev09	681	24.1K	2.3
	es-eval09	924	32.0K	3.2
	es-dev10	1,016	28.4K	2.8
	es-eval10	1,267	35.7K	3.3

A.3.2 Neural Network Training

The general training procedure for the ANN trainings for Quaero Spanish follows the general training procedure for Gale Chinese and Quaero French. A previously trained Gaussian hidden Markov model system provides the labels for the training of the recurrent and non-recurrent ANNs. Almost all ANNs are trained on phoneme classes. Triphone or context dependent states are not used on Quaero Spanish.

The multi-layer perceptrons for Spanish as well as the networks for Quaero French and Gale Chinese, are trained using the QuickNet Toolkit[3] developed at the International Computer Science Institute. The bi-directional and unidirectional RNNs and the long-short-term-memory RNNs are trained using the RNNLib[4] developed by A. Graves.

All experiments in this work on Quaero Spanish vary in the number of features, the topology and the ANN architecture used. Therefore, a general training setup for the ANN based feature extraction is hard to give. Nevertheless, the acoustic training as well as the decoding of the systems use the same concept in all experiments.

Input Features

- Short-term features: MFCC, PLP, GT
- Long-term features: temporal pattern, multi-resolution RASTA
- Global mean and variance normalization
- Temporal context: up to ± 4 frames (not included in the features)

Network Topology

- Simple and complex topologies
  - Single ANN
  - Bottle-neck structure (bottle-neck varies from 33 up to 100)
  - Hierarchical ANNs
  - Up to three hidden layers

[3] http://www.icsi.berkeley.edu/Speech/qn.html
[4] http://sourceforge.net/projects/rnnl

- Combination of feed-forward networks (multi-layer perceptrons) and RNNs

Feed-forward Network

- Up to 4000 nodes in the hidden layer

Recurrent Neural Network

- Up to 400 nodes in the hidden layer
- Bi-directional networks
- Long-short-term-memory structure

Output layer

- Phoneme classes: 33
- Softmax normalization

Others

- Performance measured on a training and a validation set
- Learning rate adjustment according to performance on the validation set
- Early stopping to avoid over-fitting

Overall, a huge number of ANN based probabilistic features is provided for the acoustic training. The ANN training is performed on 93% of the whole corpus. The other 7% of the corpus is used as validation set to measure the performance of the training and to adjust the learning rate. The random selection into training and validation set is made once for each of the two training corpora and stays unchanged for all ANN trainings.

A.3.3 Acoustic Modeling

For each feature set derived from an ANN a speaker independent and a speaker adapted acoustic model are trained using the tandem approach. The acoustic model is speaker adapted by speaker adaptive training using constrained maximum likelihood linear regression. An initial baseline system is trained using only MFCC features. The alignment derived from this initial model provides the training alignment for the ANN training as well as for the acoustic model training. In addition to the speaker independent and speaker adapted model we improve the acoustic model by discriminative training using the margin-based minimum phoneme error criterion.

The general configuration of the acoustic model is summarized by:

Input features

- Short-term features (MFCC, PLP, GT)
- Probabilistic features derived from an ANN (posterior estimates and bottle-neck features)
- Feature reduction by linear discriminant analysis of 9 adjacent input frames to 45 components

  - Short-term ($16 \times 9 = 144$)
  - ANN probabilistic features
    * Posterior estimates: $9 \times 33 = 297$
    * Bottle-neck: $9 \times \{33, 50, 75, 100\}$
- Optional: the linear discriminant analysis is exchanged by the principal component analysis to reduce the probabilistic feature derived from an ANN.
- Total feature size $45 + 45 = 90$

Acoustic Model

- $3-2$-state hidden Markov models
- Cross-word acoustic model
- State-tying via phonetic decision tree
- 4,501 mixtures with a total of 1.1M Gaussian densities

The pronunciation lexicon for Spanish is derived from the lexicon of the LC-STAR project[5]. Similar to [Bisani & Ney 03], a grapheme-to-phoneme conversion model produces the missing pronunciations.

Table A.5 Additional corpus statistics for the Spanish language.

	Testing corpora			
Corpus	es-dev10	es-eval10	es-eval09	es-dev09
perplexity	184.9	175.5	191.4	201.3

A.3.4 Decoding

The whole decoding process is divided into two passes. In the second pass the maximum likelihood trained speaker adapted acoustic model could be exchanged by the corresponding margin-based minimum phoneme error trained speaker adapted acoustic model.

- 1st decoding pass: maximum likelihood trained acoustic model using vocal tract length normalization adapted features (fast variant of vocal tract length normalization)
- 2nd decoding pass: maximum likelihood trained speaker adapted acoustic model using speaker adaptive training using constrained maximum likelihood linear regression and maximum likelihood linear regression

In recognition a 4-gram language model is used which consists of 60k words. The language model is trained on the final text editions and verbatim transcriptions of the European Parliament Plenary Sessions, and on data from the Spanish Parliament and Spanish Congress,

[5] LC-STAR: Lexica and Corpora for Speech-to-Speech Translation Components, http://www.lc-star.com

provided within the TC-STAR project [Ramabhadran & Siohan+ 06,Lööf & Gollan+ 07,Lamel & Gauvain+ 07]. In addition, the audio transcriptions as well as language model data collect within the Quaero project are used. Table A.5 summarizes the perplexity of the language model on the different corpora.

List of Figures

List of Tables

Glossary

ANN	artificial neural network
CER	character error rate
CRBE	critical band energy
GT	Gammatone
MFCC	Mel frequency cepstral coefficient
MRASTA	multi-resolution RASTA
PLP	perceptual linear prediction coefficient
RNN	recurrent neural network
SAT/CMLLR	speaker adaptive training using constrained maximum likelihood linear regression
WER	word error rate

Bibliography

[Acero 90] A. Acero: *Acoustical and Environmental Robustness in Automatic Speech Recognition*. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA, 1990.

[Aertsen & Johannesma$^+$ 80] A.M.H.J. Aertsen, P.I.M. Johannesma, D.J. Hermes: Spectro-temporal Receptive Fields of Auditory Neurons in the Grassfrog. *Biological Cybernetics*, Vol. 38, No. 4, pp. 235–248, Nov. 1980.

[Alleva & Huang$^+$ 96] P. Alleva, X.D. Huang, M.Y. Hwang: Improvements on the Pronunciation Prefix Tree Search Organization. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 1, pp. 133–136, Atlanta, GA, USA, May 1996.

[Andreou & Kamm$^+$ 94] A. Andreou, T. Kamm, J. Cohen: Experiments in Vocal Tract Normalization. In *Proc. CAIP Workshop: Frontiers in Speech Recognition II*, 1994.

[Bahl & Jelinek$^+$ 83] L.R. Bahl, F. Jelinek, R.L. Mercer: A Maximum Likelihood Approach to Continuous Speech Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 5, pp. 179–190, March 1983.

[Baker 75] J.K. Baker: Stochastic Modeling for Automatic Speech Understanding. In D.R. Reddy, editor, *Speech Recognition*, pp. 512–542. Academic Press, New York, NY, USA, 1975.

[Baum 72] L.E. Baum: An Inequality and Associated Maximization Technique in Statistical Estimation for Probabilistic Functions of Markov Processes. In O. Shisha, editor, *Inequalities*, Vol. 3, pp. 1–8. Academic Press, New York, NY, 1972.

[Bayes 63] T. Bayes: An Essay Towards Solving a Problem in the Doctrine of Chances. *Philosophical Transactions of the Royal Society of London*, Vol. 53, pp. 370–418, 1763. Reprinted in *Biometrika*, vol. 45, no. 3/4, pp. 293–315, December 1958.

[Bellman 57] R.E. Bellman: Dynamic Programming. Princeton University Press, Princeton, NJ, USA, 1957.

[Bengio 09] Y. Bengio: Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, Vol. 2, No. 1, pp. 1–127, 2009.

[Bengio & Ducharme 01] Y. Bengio, R. Ducharme: A neural probabilistic language model. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 932–938, Vancouver, BC, Canada, Dec. 2001.

[Bengio & Lamblin$^+$ 06] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle: Greedy Layer-wise Training of Deep Networks. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 153–160, Vancouver, B.C., Canada, Dec. 2006.

[Bengio & Simard$^+$ 94] Y. Bengio, P. Simard, P. Frasconi: Learning Long-term Dependencies with Gradient Descent is Difficult. *IEEE Transactions on Neural Networks*, Vol. 5, No. 2, pp. 157–166, Sept. 1994.

[Bertolami & Bunke 08] R. Bertolami, H. Bunke: Hidden Markov Model-based Ensemble Methods for Offline Handwritten Text Line Recognition. *Pattern Recognition*, Vol. 41, pp. 3452–3460, Nov. 2008.

[Beulen 99] K. Beulen: *Phonetische Entscheidungsbäume für die automatische Spracherkennung mit großem Vokabular*. Ph.D. thesis, Human Language Technology and Pattern Recognition Group, RWTH Aachen University, Aachen, Germany, July 1999.

[Beulen & Welling$^+$ 95] K. Beulen, L. Welling, H. Ney: Experiments with Linear Feature Extraction in Speech Recognition. In *European Conference on Speech Communication and Technology (Eurospeech)*, pp. 1415–1418, Madrid, Spain, Sept. 1995.

[Bisani & Ney 03] M. Bisani, H. Ney: Multigram-based Grapheme-to-Phoneme Conversation for LVCSR. In *Interspeech*, pp. 933–936, Geneva, Switzerland, Sept. 2003.

[Bishop 96] C.M. Bishop: *Neural Networks for Pattern Recognition*. Oxford University Press, USA, 1 edition, Jan. 1996.

[Boquera & Bleda$^+$ 11] S.E. Boquera, M.J.C. Bleda, J.G. Moya, F.Z. Martinez: Improving Offline Handwritten Text Recognition with Hybrid HMM/ANN Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 33, pp. 767–779, April 2011.

[Bourland & Wellekens 87] H. Bourland, C.J. Wellekens: Multi-layer Perceptron and Automatic Speech Recognition. In *International Conference on Neural Networks (ICNN)*, San Diego, California, June 1987.

[Bourlard & Morgan 93] H.A. Bourlard, N. Morgan: *Connectionist Speech Recognition: A Hybrid Approach*. Kluwer Academic Publishers, Norwell, MA, USA, 1993.

[Cardinal & Dumouchel$^+$ 08] P. Cardinal, P. Dumouchel, G. Boulianne, M. Comeau: GPU accelerated acoustic likelihood computations. In *Interspeech*, pp. 964–967, Brisbane, Australia, Sept. 2008.

[Chen & Chang+ 03] B. Chen, S. Chang, S. Sivadas: Learning Discriminative Temporal Patterns in Speech: Development of Novel TRAPS-Like Classifiers. In *Interspeech*, pp. 853–856, Geneva, Switzerland, Sept. 2003.

[Chen & Gopalakrishnan 98] S.S. Chen, P.S. Gopalakrishnan: Clustering via the Bayesian Information Criterion with Applications in Speech Recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 2, pp. 645–648, Seattle, Washington, USA, March 1998.

[Chen & Gopinath+ 97] C.J. Chen, R.A. Gopinath, M.D. Monkowski, M.A. Picheny, K. Shen: New Methods in Continuous Mandarin Speech Recognition. In *European Conference on Speech Communication and Technology (Eurospeech)*, Vol. 3, pp. 1543–1546, Rhodes, Greece, Sept. 1997.

[Chen & Zhu+ 04] B. Chen, Q. Zhu, N. Morgan: Learning Long-term Temporal Features in LVCSR using Neural Networks. In *Interspeech*, pp. 612–615, Jeju Island, Korea, Oct. 2004.

[Chu & KuoZhang+ 08] S.M. Chu, H.k. KuoZhang, L. Mangu, Y. Liu, Y. Qin, Q. Shi, S.L. Zhang, H. Aronowitz: Recent Advantages in the GALE Mandarin Transcription System. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 4329–4333, Las Vegas, NV, USA, April 2008.

[Davis & Mermelstein 80] S. Davis, P. Mermelstein: Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. 28, No. 4, pp. 357–366, Aug. 1980.

[Dempster & Laird+ 77] A. Dempster, N. Laird, D. Rubin: Maximum Lkelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society*, Vol. 39, No. 2, pp. 1–38, 1977.

[Dijkstra 59] E.W. Dijkstra: A Note on Two Problems in Connection with Graphs. *Numerische Mathematik*, Vol. 1, pp. 269–271, 1959.

[Doddington & Przybocki+ 00] G.R. Doddington, M.A. Przybocki, A.F. Martin, D.A. Reynolds: The NIST Speaker Recognition Evaluation – Overview, Methodology, Systems, Results, Perspective. *Speech Communication*, Vol. 31, No. 2-3, pp. 225–254, June 2000.

[Dötsch 11] P. Dötsch: Optimization of Hidden Markov Models and Neural Networks. Master's thesis, RWTH Aachen University, Aachen, Germany, Dec. 2011.

[Dreuw 12] P. Dreuw: *Probabilistic Sequence Models for Image Sequence Processing and Recognition*. Ph.D. thesis, RWTH Aachen University, Aachen, Germany, April 2012.

[Dreuw & Deselaers+ 06] P. Dreuw, T. Deselaers, D. Rybach, D. Keysers, H. Ney: Tracking Using Dynamic Programming for Appearance-Based Sign Language Recognition. In *IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 293–298, Southampton, UK, April 2006.

[Dreuw & Dötsch+ 11] P. Dreuw, P. Dötsch, C. Plahl, H. Ney: Hierarchical Hybrid MLP/HMM or rather MLP Features for a Discriminatively Trained Gaussian HMM: A Comparison for Offline Handwriting Recognition. In *IEEE International Conference on Image Processing (ICIP)*, pp. 3602–3605, Brussels, Belgium, Sept. 2011.

[Dreuw & Heigold+ 09] P. Dreuw, G. Heigold, H. Ney: Confidence-Based Discriminative Training for Model Adaptation in Offline Arabic Handwriting Recognition. In *International Conference on Document Analysis and Recognition (ICDAR)*, pp. 596–600, Barcelona, Spain, July 2009.

[Dreuw & Heigold+ 11] P. Dreuw, G. Heigold, H. Ney: Confidence and Margin-Based MMI/MPE Discriminative Training for Offline Handwriting Recognition. *International Journal on Document Analysis and Recognition*, Vol. 14, No. 3, pp. 273–288, April 2011.

[Dreuw & Jonas+ 08] P. Dreuw, S. Jonas, H. Ney: White-Space Models for Offline Arabic Handwriting Recognition. In *International Conference on Pattern Recognition (ICPR)*, pp. 1–4, Tampa, Florida, USA, Dec. 2008.

[Duda & Hart+ 01] R.O. Duda, P.E. Hart, D.G. Stork: *Pattern Classification*. Wiley-Interscience, 2 edition, Nov. 2001.

[Dunne 07] R.A. Dunne: *A Statistical Approach to Neural Networks for Pattern Recognition (Wiley Series in Computational Statistics)*. Wiley-Interscience, 2007.

[Elman 90] J.L. Elman: Finding Structure in Time. *Cognitive Science*, Vol. 14, No. 2, pp. 179–211, 1990.

[Erhan & Courville+ 10] D. Erhan, A. Courville, Y. Bengio, P. Vincent: Why does Unsupervised Pre-training Help Deep Learning. *Journal of Machine Learning Research*, Vol. 11, pp. 625–660, 2010.

[Espana-Boquera & Castro-Bleda+ 11] S. Espana-Boquera, M.J. Castro-Bleda, J. Gorbe-Moya, F. Zamora-Martinez: Improving Offline Handwritten Text Recognition with Hybrid HMM/ANN Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 33, pp. 767–779, 2011.

[Evermann & Woodland 00] G. Evermann, P. Woodland: Posterior Probability Decoding, Confidence Estimation and System Combination. In *NIST Speech Transcription Workshop*, College Park, MD, March 2000.

[Fink 03] G.A. Fink: *Mustererkennung mit Markov-Modellen*. Leitfäden der Informatik. Teubner B.G. GmbH, Stuttgart – Leipzig – Wiesbaden, Oct. 2003.

[Fiscus 97] J.G. Fiscus: A Post-Processing System to Yield Reduced Word Error Rates: Recognizer Output Voting Error Reduction (ROVER). In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 347–354, Santa Barbara, CA, USA, Dec. 1997.

[Fousek 07] P. Fousek: *Extraction of Features for Automatic Recognition of Speech Based on Spectral Dynamics*. Ph.D. thesis, Czech Technical University in Prague, Prague, Czech, March 2007.

[Gales 98] M.J.F. Gales: Maximum Likelihood Linear Transformations for HMM-Based Speech Recognition. *Computer Speech and Language*, Vol. 12, pp. 75–98, 1998.

[Generet & Ney$^+$ 95] M. Generet, H. Ney, F. Wessel: Extensions to Absolute Discounting for Language Modeling. In *European Conference on Speech Communication and Technology (Eurospeech)*, Vol. 2, pp. 1245–1248, Madrid, Spain, Sept. 1995.

[Giménez & Khoury$^+$ 10] A. Giménez, I. Khoury, A. Juan: Windowed Bernoulli Mixture HMMs for Arabic Handwritten Word Recognition. In *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 533–538, Kolkata, India, Nov. 2010.

[Graves & Bunke$^+$ 07] A. Graves, H. Bunke, S. Fernández, M. Liwicki, J. Schmidhuber: Unconstrained Online Handwriting Recognition with Recurrent Neural Networks. In *Advances in Neural Information Processing Systems (NIPS)*, Vancouver, B.C., Canada, Dec. 2007.

[Graves & Liwicki$^+$ 09] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, J. Schmidhuber: A Novel Connectionist System for Unconstrained Handwriting Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 31, No. 5, pp. 855–868, May 2009.

[Graves & Schmidhuber 08] A. Graves, J. Schmidhuber: Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 545–552, Vancouver, B.C., Canada, Dec. 2008.

[Greenwood 90] D.D. Greenwood: A Cochlear Frequency-position Function for Several Species – 29 years later. *Acoustical Society of America Journal*, Vol. 87, pp. 2592–2605, June 1990.

[Grézl & Karafiat$^+$ 07] F. Grézl, M. Karafiat, S. Kontar, J. Cernock: Probabilistic and Bottle-Neck Features for LVCSR of Meetings. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 4, pp. 757–760, Honolulu, HI, USA, April 2007.

[Grézl & Karafiát$^+$ 11] F. Grézl, M. Karafiát, M. Janda: Study of Probabilistic and Bottle-Neck Features in Multilingual Environment. In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 359–364, Hawaii, USA, Dec. 2011.

[Guiliani & Brugnara 06] D. Guiliani, F. Brugnara: Acoustic Model Adaptation with Multiple Supervisions. In *Proc.* TC-STAR *Workshop on Speech-to-Speech Translation*, pp. 151–154, Barcelona, Spain, June 2006.

[Gweth & Plahl$^+$ 12] Y. Gweth, C. Plahl, H. Ney: Enhanced Continuous Sign Language Recognition using PCA and Neural Network Features. In *CVPR 2012 Workshop on Gesture Recognition*, pp. 55–60, Providence, Rhode Island, June 2012.

[Häb-Umbach & Ney 94] R. Häb-Umbach, H. Ney: Improvements in Beam Search for 10000-Word Continuous-Speech Recognition. *IEEE Transactions on Speech and Audio Processing*, Vol. 2, No. 2, pp. 353–356, April 1994.

[Haeb-Umbach & Ney 92] R. Haeb-Umbach, H. Ney: Linear Discriminant Analysis for Improved Large Vocabulary Continuous Speech Recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 1, pp. 13–16, March 1992.

[Heigold 10] G. Heigold: *A Log-Linear Discriminative Modeling Framework for Speech Recognition*. Ph.D. thesis, RWTH Aachen University, Aachen, Germany, June 2010.

[Heigold & Deselaers$^+$ 08] G. Heigold, T. Deselaers, R. Schlüter, H. Ney: Modified MMI/MPE: A Direct Evaluation of the Margin in Speech Recognition. In *International Conference on Machine Learning (ICML)*, pp. 384–391, Helsinki, Finland, July 2008.

[Hermansky 90] H. Hermansky: Perceptual Linear Prediction (PLP) Analysis for Speech. *Journal of the Acoustical Society of America*, Vol. 87, No. 4, pp. 1738–1752, June 1990.

[Hermansky & Ellis$^+$ 00] H. Hermansky, D. Ellis, S. Sharma: Tandem Connectionist Feature Stream Extraction for Conventional HMM Systems. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 1635–1638, June 2000.

[Hermansky & Fousek 05] H. Hermansky, P. Fousek: Multi-resolution RASTA filtering for TANDEM-based ASR. In *Interspeech*, pp. 361–364, Lisbon, Portugal, Sept. 2005.

[Hermansky & Sharma 98] H. Hermansky, S. Sharma: TRAPs - Classifiers of Temporal Patterns. In *International Conference on Spoken Language Processing (ICSLP)*, pp. 1003–1006, Nov. 1998.

[Hillard & Hoffmeister$^+$ 07] D. Hillard, B. Hoffmeister, M. Ostendorf, R. Schlüter, H. Ney: iROVER: Improving System Combination with Classification. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pp. 65–68, Rochester, New York, April 2007.

[Hinton 02] G.E. Hinton: Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, Vol. 14, pp. 1771–1800, Aug. 2002.

[Hinton 10] G. Hinton: A Practical Guide to Training Restricted Boltzmann Machines. Technical Report UTML TR 2010-003, University of Toronto, 2010.

[Hinton & Osindero$^+$ 06] G.E. Hinton, S. Osindero, Y.W. Teh: A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, Vol. 18, No. 7, pp. 1527–1554, July 2006.

[Hinton & Salakhutdinov 06] G.E. Hinton, R.R. Salakhutdinov: Reducing the Dimensionality of Data with Neural Networks. *Science (New York, N.Y.)*, Vol. 313, No. 5786, pp. 504–507, July 2006.

[Hochreiter & Bengio$^+$ 01] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber: Gradient Flow in Recurrent Nets: the Difficulty of Learning Long-term Dependencies. In *A Field Guide to Dynamical Recurrent Neural Networks*. IEEE Press, 2001.

[Hochreiter & Schmidhuber 97] S. Hochreiter, J. Schmidhuber: Long Short-Term Memory. *IEEE Transactions on Neural Networks*, Vol. 9, No. 8, pp. 1735–1780, Nov. 1997.

[Hoffmeister 11] B. Hoffmeister: *Bayes Risk Decoding and its Application to System Combination*. Ph.D. thesis, RWTH Aachen University, Aachen, Germany, July 2011.

[Hoffmeister & Plahl$^+$ 07] B. Hoffmeister, C. Plahl, P. Fritz, G. Heigold, J. Lööf, R. Schlüter, H. Ney: Development of the 2007 RWTH Mandarin LVCSR System. In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 455–460, Kyoto, Japan, Dec. 2007.

[Hoffmeister & Schlüter$^+$ 08] B. Hoffmeister, R. Schlüter, H. Ney: iCNC and iROVER: The Limits of Improving System Combination with Classification? In *Interspeech*, pp. 232–235, Brisbane, Australia, Sept. 2008.

[Hopfield 82] J.J. Hopfield: Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proceedings of the National Academy of Sciences*, Vol. 79, No. 8, pp. 2554–2558, April 1982.

[Huang & Jack 89] X.D. Huang, M.A. Jack: Semi-Continuous Hidden Markov Models for Speech Signals. *Computer Speech and Language*, Vol. 3, No. 3, pp. 329–252, 1989.

[Hwang & Peng$^+$ 07] M.Y. Hwang, G. Peng, W. Wang, A. Faria, A. Heidel, M. Ostendorf: Building a Highly Accurate Mandarin Speech Recognizer. In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 490–495, Kyoto, Japan, Dec. 2007.

[Jelinek 69] F. Jelinek: A Fast Sequential Decoding Algorithm Using a Stack. *IBM Journal of Research and Development*, Vol. 13, pp. 675–685, Nov. 1969.

[Jelinek 76] F. Jelinek: Continuous Speech Recognition by Statistical Methods. *Proceedings of the IEEE*, Vol. 64, No. 10, pp. 532–556, April 1976.

[Jordan 89] M.I. Jordan: Serial Order: A Parallel, Distributed Processing Approach. In *Advances in Connectionist Theory: Speech*. Erlbaum, Hillsdale, NJ, 1989.

[Kanthak & Schütz$^+$ 00] S. Kanthak, K. Schütz, H. Ney: Using SIMD Instructions for Fast Likelihood Calculation in LVCSR. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 1531–1534, Istanbul, Turkey, June 2000.

[Katz 87] S.M. Katz: Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Transactions on Speech and Audio Processing*, Vol. 35, pp. 400–401, March 1987.

[Kneser & Ney 95] R. Kneser, H. Ney: Improved Backing-off for M-gram Language Modeling. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 1, pp. 181–184, Detroit, Michigan, USA, May 1995.

[Lamel & Gauvain$^+$ 07] L. Lamel, J.L. Gauvain, G. Adda, C. Barras, E. Bilinski, O. Galibert, A. Pujol, H. Schwenk, X. Zhu: The LIMSI 2006 TC-STAR EPPS Transcription Systems. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 997–1000, Honolulu, HI, USA, April 2007.

[Lee & Rose 96] L. Lee, R.C. Rose: Speaker Normalization using Efficient Frequency Warping Procedures. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 353–356, Atlanta, GA, USA, May 1996.

[Lee & Rose 98] L. Lee, R. Rose: A Frequency Warping Approach to Speaker Normalization. *IEEE Transactions on Speech and Audio Processing*, Vol. 6, No. 1, pp. 49–60, Jan. 1998.

[Leggetter & Woodland 95] C.J. Leggetter, P.C. Woodland: Maximum Likelihood Linear Regression for Speaker Adaptation of Continuous Density Hidden Markov Models. *Computer Speech and Language*, Vol. 9, No. 2, pp. 171–185, 1995.

[Lei & Siu$^+$ 06] X. Lei, M. Siu, M.Y. Hwang, M. Ostendorf, T. Lee: Improved Tone Modeling for Mandarin Broadcast News Speech Recognition. In *Interspeech*, pp. 1237–1240, Pittsburgh, Pennsylvania, USA, Sept. 2006.

[Lei & Wu$^+$ 09] X. Lei, W. Wu, W. Wang, A. Mandal, A. Stolcke: Development of the 2008 SRI Mandarin Speech-to-Text System for Broadcast News and Conversation. In *Interspeech*, pp. 2099–2102, Brighton, U.K., Sept. 2009.

[Levinson & Rabiner$^+$ 83] S.E. Levinson, L.R. Rabiner, M.M. Sondhi: An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition. *Bell System Technical Journal*, Vol. 62, No. 4, pp. 1035–1074, April 1983.

[Lippmann 89] R. Lippmann: Review of Neural Networks for Speech Recognition. *Neural Computation*, Vol. 1, pp. 1–38, 1989.

[Liwicki & Bunke 05] M. Liwicki, H. Bunke: IAM-OnDB - an On-Line English Sentence Database Acquired from Handwritten Text on a Whiteboard. In *International Conference on Document Analysis and Recognition (ICDAR)*, pp. 956–961, Seoul, Korea, Aug. 2005.

[Ljolje & Pereira$^+$ 99] A. Ljolje, F. Pereira, M. Riley: Efficient General Lattice Generation and Rescoring. In *European Conference on Speech Communication and Technology (Eurospeech)*, pp. 1251–1254, Budapest, Hungary, Sept. 1999.

[Lööf & Gollan$^+$ 07] J. Lööf, C. Gollan, S. Hahn, G. Heigold, B. Hoffmeister, C. Plahl, D. Rybach, R. Schlüter, H. Ney: The RWTH 2007 TC-STAR Evaluation System for European English and Spanish. In *Interspeech*, pp. 2145–2148, Antwerp, Belgium, Aug. 2007.

[Lowerre 76] B. Lowerre: *A Comparative Performance Analysis of Speech Understanding Systems*. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA, 1976.

[Märgner & Abed 09] V. Märgner, H.E. Abed: ICDAR 2009 Arabic Handwriting Recognition Competition. In *International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1383–1387, Barcelona, Spain, July 2009.

[Märgner & Abed 10] V. Märgner, H.E. Abed: ICFHR 2010 - Arabic Handwriting Recognition Competition. In *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 709–714, Kolkata, India, Nov. 2010.

[Mikolov & Karafiàt$^+$ 10] T. Mikolov, M. Karafiàt, L. Burget, J. Černockỳ, S. Khudanpur: Recurrent neural network based language model. In *Interspeech*, pp. 1045–1048, Makuhari, Japan, Sept. 2010.

[Mikolov & Kombrink$^+$ 11] T. Mikolov, S. Kombrink, L. Burget, J. Černockỳ, S. Khudanpur: Extensions of Recurrent Neural Network Language Model. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 5528–5531, Prague, Czech Republic, May 2011.

[Mohamed & Dahl$^+$ 09] A.r. Mohamed, G.E. Dahl, G.E. Hinton: Deep Belief Networks for Phone Recognition. In *NIPS Workshop on Deep Learning for Speech Recognition and Related Applications*, Whistler (BC), Canada, Dec. 2009.

[Mohamed & Sainath$^+$ 11] A.r. Mohamed, T.N. Sainath, G. Dahl, B. Ramabhadran, G.E. Hinton, M.A. Picheny: Deep Belief Networks using Discriminative Features for Phone Recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 5060–5063, Prague, Czech Republic, May 2011.

[Mohamed & Yu$^+$ 10] A.r. Mohamed, D. Yu, L. Deng: Investigation of Full-Sequence Training of Deep Belief Networks for Speech Recognition. In *Interspeech*, pp. 1692–1695, Makuhari, Japan, Sept. 2010.

[Murveit & Butzberger$^+$ 93] H. Murveit, J. Butzberger, V. Digalakis, M. Weintraub: Progressive-search algorithms for large-vocabulary speech recognition. In *Proceedings of the workshop on Human Language Technology (HLT)*, pp. 87–90, Morristown, NJ, USA, Jan. 1993. Association for Computational Linguistics.

[Ney 84] H. Ney: The Use of a One-Stage Dynamic Programming Algorithm for Connected Word Recognition. *IEEE Transactions on Speech and Audio Processing*, Vol. 32, No. 2, pp. 263–271, April 1984.

[Ney 90] H. Ney: Acoustic Modeling of Phoneme Units for Continuous Speech Recognition. In *Signal Processing V: Theories and Applications, Fifth European Signal Processing Conference*, pp. 65–72. Elsevier Science Publishers B. V., Barcelona, Spain, 1990.

[Ney & Aubert 94] H. Ney, X. Aubert: A Word Graph Algorithm for Large Vocabulary Continuous Speech Recognition. In *International Conference on Spoken Language Processing (ICSLP)*, Vol. 3, pp. 1355–1358, Yokohama, Japan, Sept. 1994.

[Ney & Essen$^+$ 94] H. Ney, U. Essen, R. Kneser: On Structuring Probabilistic Dependencies in Language Modeling. *Computer Speech and Language*, Vol. 2, No. 8, pp. 1–38, 1994.

[Ney & Martin$^+$ 97] H. Ney, S.C. Martin, F. Wessel: Statistical Language Modeling using Leaving-One-Out. In S. Young, G. Bloothooft, editors, *Corpus Based Methods in Language and Speech Processing*, pp. 1–26. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1997.

[Ney & Mergel$^+$ 87] H. Ney, D. Mergel, A. Noll, A. Paeseler: A Data-Driven Organization of the Dynamic Programming Beam Search for Continuous Speech Recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 833–836, Dallas, TX, USA, April 1987.

[Ng & Zhang$^+$ 08] T. Ng, B. Zhang, K. Nguyen, L. Nguyen: Progress in the BBN Mandarin Speech to Text System. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 1537–1540, Las Vegas, NV, USA, April 2008.

[Nolden & Ney$^+$ 11] D. Nolden, H. Ney, R. Schlüter: Exploiting Sparseness of Backing-Off Language Models for Efficient Look-Ahead in LVCSR. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 4684–4687, Prague, Czech Republic, May 2011.

[Nolden & Schlüter$^+$ 11] D. Nolden, R. Schlüter, H. Ney: Acoustic Look-Ahead for More Efficient Decoding in LVCSR. In *Interspeech*, pp. 893–896, Florence, Italy, Aug. 2011.

[Nußbaum-Thom & Wiesler$^+$ 10] M. Nußbaum-Thom, S. Wiesler, M. Sundermeyer, C. Plahl, S. Hahn, R. Schlüter, H. Ney: The RWTH 2009 Quaero ASR Evaluation System for English and German. In *Interspeech*, pp. 1517–1520, Makuhari, Japan, Sept. 2010.

[Ortmanns & Ney$^+$ 96] S. Ortmanns, H. Ney, A. Eiden, N. Coenen: Look-Ahead Techniques for Improved Beam Search. In *CRIM-FORWISS Workshop*, pp. 10–22, Montreal, Canada, Oct. 1996.

[Ortmanns & Ney$^+$ 97a] S. Ortmanns, H. Ney, T. Firzlaff: Fast Likelihood Computation Methods for Continuous Mixture Densities in Large Vocabulary Speech Recognition. In *European Conference on Speech Communication and Technology (Eurospeech)*, Vol. 1, pp. 139–142, Rhodes, Greece, Sept. 1997.

[Ortmanns & Ney$^+$ 97b] S. Ortmanns, H. Ney, X. Aubert: A Word Graph Algorithm for Large Vocabulary Continuous Speech Recognition. *Computer Speech and Language*, Vol. 11, No. 1, pp. 43–72, Jan. 1997.

[Parihar & Schlüter$^+$ 09] N. Parihar, R. Schlüter, D. Rybach, E.A. Hansen: Parallel Fast Likelihood Computation for LVCSR using Mixture Decomposition. In *Interspeech*, pp. 3047–3050, Brighton, U.K., Sept. 2009.

[Paul 91] D.B. Paul: Algorithms for an Optimal A^* Search and Linearizing the Search in the Stack Decoder. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 1, pp. 693–696, Toronto, Canada, May 1991.

[Pechwitz & Maddouri$^+$ 02] M. Pechwitz, S.S. Maddouri, V. Mägner, N. Ellouze, H. Amiri: IFN/ENIT-database of Handwritten Arabic Words. In *Colloque International Francophone sur l Ecrit et le Document (CIFED)*, pp. 129–136, Hammamet, Tunis, Oct. 2002.

[Peeling & Moore$^+$ 86] S.M. Peeling, R.K. Moore, M.J. Tomlinson: The Multi-layer Perceptron as a Tool For Speech Pattern Processing Research. In *Proc. Institute of Acoustics, Autumn Conference on Speech and Hearing*, Vol. 8, pp. 307–314, Windermere, Nov. 1986. Institute of Acoustics, Edinburgh.

[Pitz 05] M. Pitz: *Investigations on Linear Transformations for Speaker Adaptation and Normalization*. Ph.D. thesis, RWTH Aachen University, Aachen, Germany, March 2005.

[Plahl & Hoffmeister$^+$ 08] C. Plahl, B. Hoffmeister, M.Y. Hwang, D. Lu, G. Heigold, J. Lööf, R. Schlüter, H. Ney: Recent Improvements of the RWTH GALE Mandarin LVCSR System. In *Interspeech*, pp. 2426–2429, Brisbane, Australia, Sept. 2008.

[Plahl & Hoffmeister$^+$ 09] C. Plahl, B. Hoffmeister, G. Heigold, J. Lööf, R. Schlüter, H. Ney: Development of the GALE 2008 Mandarin LVCSR System. In *Interspeech*, pp. 2107–2110, Brighton, England, Sept. 2009.

[Plahl & Sainath$^+$ 12] C. Plahl, T.N. Sainath, B. Ramabhadran, D. Nahamoo: Improved Pretraining of Deep Belief Networks using Sparse Encoding Symmetric Machines. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 4165–4168, Kyoto, Japan, March 2012.

[Plahl & Schlüter$^+$ 10] C. Plahl, R. Schlüter, H. Ney: Hierarchical Bottle Neck Features for LVCSR. In *Interspeech*, pp. 1197–1200, Makuhari, Japan, Sept. 2010.

[Plahl & Schlüter$^+$ 11a] C. Plahl, R. Schlüter, H. Ney: Cross-lingual Portability of Chinese and English Neural Network Features for French and German LVCSR. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 371–376, Hawaii, USA, Dec. 2011.

[Plahl & Schlüter$^+$ 11b] C. Plahl, R. Schlüter, H. Ney: Improved Acoustic Feature Combination for LVCSR by Neural Networks. In *Interspeech*, pp. 1237–1240, Florence, Italy, Aug. 2011.

[Povey & Kanevsky$^+$ 08] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, K. Visweswariah: Boosted MMI for Model and Feature-space Discriminative Training. In

IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 4057–4060, Las Vegas, NV, USA, April 2008.

[Povey & Woodland 02] D. Povey, P.C. Woodland: Minimum Phone Error and I-Smoothing for Improved Discriminative Training. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 1, pp. 105–108, Orlando, FL, USA, May 2002.

[Qian & Xu$^+$ 11] Y. Qian, J. Xu, D. Povey, J. Liu: Strategies for Using MLP Based Features with Limited Target-language Training Data. In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 354–358, Hawaii, USA, Dec. 2011.

[Rabiner & Juang 86] L. Rabiner, B.H. Juang: An Introduction to Hidden Markov Models. *IEEE ASSP Magazine*, Vol. 3, No. 1, pp. 4–16, 1986.

[Rabiner & Schafer 79] L.R. Rabiner, R.W. Schafer: *Digital Processing of Speech Signals*. Prentice-Hall Signal Processing Series, Englewood Cliffs, NJ, 1979.

[Ramabhadran & Siohan$^+$ 06] B. Ramabhadran, O. Siohan, L. Mangu, G. Zweig, M. Westphal, H. Schulz, A. Soneiro: The IBM 2006 Speech Transcription System for European Parliamentary Speeches. In *Interspeech*, pp. 1225–1228, Pittsburgh, PA, USA, Sept. 2006.

[Ramasubramansian & Paliwal 92] V. Ramasubramansian, K.K. Paliwal: Fast k-dimensional Tree Algorithms for Nearest Neighbor Search with Application to Vector Quantization Encoding. *IEEE Transactions on Speech and Audio Processing*, Vol. 40, No. 3, pp. 518–528, March 1992.

[Ranzato & Boureau$^+$ 07a] M. Ranzato, Y.L. Boureau, S. Chopra, Y. LeCun: A Unified Energy-Based Framework for Unsupervised Learning. *Journal of Machine Learning Research - Proceedings Track*, Vol. 2, pp. 371–379, 2007.

[Ranzato & Boureau$^+$ 07b] M. Ranzato, Y.L. Boureau, Y. LeCun: Sparse Feature Learning for Deep Belief Networks. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 1137–1144, Vancouver, B.C., Canada, Dec. 2007.

[Ranzato & Poultney$^+$ 06] M.A. Ranzato, C.S. Poultney, S. Chopra, Y. LeCun: Efficient Learning of Sparse Representations with an Energy-Based Model. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 1137–1144, Vancouver, B.C., Canada, Dec. 2006.

[Reed & Marks 99] R.D. Reed, R.J. Marks: *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks*. MIT Press, 1999.

[Robinson & Fallside 87a] A.J. Robinson, F. Fallside: Static and Dynamic Error Propagation Networks with Application to Speech Coding. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 632–641, Denver, CO, USA, Dec. 1987.

[Robinson & Fallside 87b] A.J. Robinson, F. Fallside: The Utility Driven Dynamic Error Propagation Network. Technical Report CUED/F-INFENG/TR.1, Cambridge University Engineering Department, Cambridge, 1987.

[Rumelhart & Hinton$^+$ 86] D.E. Rumelhart, G.E. Hinton, R.J. Williams: Learning Internal Representations by Error Propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 1, pp. 318–362. MIT Press, Cambridge, MA, USA, 1986.

[Rumelhart & Hinton$^+$ 88] D.E. Rumelhart, G.E. Hinton, R.J. Williams: Learning Internal Representations by Error Propagation. In J.A. Anderson, E. Rosenfeld, editors, *Neurocomputing: foundations of research*, pp. 673–695. MIT Press, Cambridge, MA, USA, 1988.

[Rybach & Gollan$^+$ 09] D. Rybach, C. Gollan, G. Heigold, B. Hoffmeister, J. Lööf, R. Schlüter, H. Ney: The RWTH Aachen University Open Source Speech Recognition System. In *Interspeech*, pp. 2111–2114, Brighton, UK, Sept. 2009.

[Sainath & Kingsbury$^+$ 11] T.N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novak: Making Deep Belief Networks Effective for Large Vocabulary Continuous Speech Recognition. In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 30–35, Hawaii, USA, Dec. 2011.

[Sakoe 79] H. Sakoe: Two-Level DP-Matching - A Dynamic Programming-Based Pattern Matching Algorithm for Connected Word Recognition. *IEEE Transactions on Speech and Audio Processing*, Vol. 27, pp. 588–595, Dec. 1979.

[Salakhutdinov 09] R. Salakhutdinov: *Learning Deep Generative Models*. Ph.D. thesis, University of Toronto, Toronto, Canada, 2009.

[Salakhutdinov & Hinton 09] R. Salakhutdinov, G.E. Hinton: Deep Boltzmann Machines. *Journal of Machine Learning Research - Proceedings Track*, Vol. 5, pp. 448–455, 2009.

[Salakhutdinov & Larochelle 10] R. Salakhutdinov, H. Larochelle: Efficient Learning of Deep Boltzmann Machines. *Journal of Machine Learning Research - Proceedings Track*, Vol. 9, pp. 693–700, 2010.

[Salakhutdinov & Murray 08] R. Salakhutdinov, I. Murray: On the Quantitative Analysis of Deep Belief Networks. In *International Conference on Machine Learning (ICML)*, pp. 872–879, Helsinki, Finland, July 2008.

[Sanand & Schlüter$^+$ 10] D.R. Sanand, R. Schlüter, H. Ney: Revisiting VTLN Using Linear Transformation on Conventional MFCC. In *Interspeech*, pp. 538–541, Makuhari, Japan, Sept. 2010.

[Schenk & Rigoll 06] J. Schenk, G. Rigoll: Novel Hybrid NN/HMM Modellibengio2007:gltong Techniques for On-line Handwriting Recognition. In *International Workshop on Frontiers in Handwriting Recognition (IWFHR)*, La Baule, France, Oct. 2006.

[Schlüter 00] R. Schlüter: *Investigations on Discriminative Training Criteria*. Ph.D. thesis, RWTH Aachen University, Aachen, Germany, Sept. 2000.

[Schlüter & Bezrukov$^+$ 07] R. Schlüter, I. Bezrukov, H. Wagner, H. Ney: Gammatone Features and Feature Combination for Large Vocabulary Speech Recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 649–652, Honululu, HI, USA, April 2007.

[Schlüter & Zolnay$^+$ 06] R. Schlüter, A. Zolnay, H. Ney: Feature Combination using Linear Discriminant Analysis and Its Pitfalls. In *Interspeech*, pp. 345–348, Pittsburgh, PA, USA, Sept. 2006.

[Schultz 02] T. Schultz: GlobalPhone: A Multilingual Speech and Text Database Developed at Karlsruhe University. In *Interspeech*, pp. 345–348, Denver, CO, USA, Sept. 2002.

[Schuster & Paliwal 97] M. Schuster, K.K. Paliwal: Bidirectional Recurrent Neural Networks. *IEEE Transactions on Signal Processing*, Vol. 45, No. 11, pp. 2673–2681, Nov. 1997.

[Schwartz & Chow 90] R. Schwartz, Y.L. Chow: The *N*-Best Algorithm: An Efficient and Exact Procedure for Finding the *N* most likely Sentence Hypotheses. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 81–84, Albuquerque, NM, April 1990.

[Schwarz & Matejka$^+$ 06] P. Schwarz, P. Matejka, J. Cernocky: Hierarchical Structures of Neural Networks for Phoneme Recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 325–328, Toulouse, France, May 2006.

[Schwenk & Gauvain 02] H. Schwenk, J.L. Gauvain: Connectionist Language Modeling for Large Vocabulary Continuous Speech Recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 765–768, Orlando, FL, USA, May 2002.

[Seide & Gang$^+$ 11] F. Seide, L. Gang, Y. Dong: Conversational Speech Transcription using context-dependent Deep Neural Network. In *Interspeech*, pp. 437–440, Florence, Italy, Aug. 2011.

[Seide & Li$^+$ 11] F. Seide, G. Li, X. Chen, D. Yu: Feature engineering in Context-Dependent Deep Neural Networks for Conversational Speech Transcription. In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 24–29, Hawaii, USA, Dec. 2011.

[Sivadas & Hermansky 02] S. Sivadas, H. Hermansky: Hierarchical Tandem Feature Extraction. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 809–812, Orlando, FL, USA, May 2002.

[Stemmer & Brugnara$^+$ 05] G. Stemmer, F. Brugnara, D. Giuliani: Adaptive Training Using Simple Target Models. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 1, pp. 997–1000, Philadelphia, PA, March 2005.

[Stolcke 02] A. Stolcke: SRILM - an Extensible Language Modeling Toolkit. In *Interspeech*, pp. 901–904, Denver, CO, USA, Sept. 2002.

[Stolcke & Bratt+ 00] A. Stolcke, H. Bratt, J. Butzberger, H. Franco, V.R.R. Gadde, M. Plauche, C. Richey, E. Shriberg, K. Sönmez, F. Weng, J. Zheng: The SRI March 2000 Hub-5 Conversational Speech Transcription System. In *NIST Speech Transcription Workshop*, College Park, MD, USA, May 2000.

[Stolcke & Grézl+ 06] A. Stolcke, F. Grézl, M.Y. Hwang, X. Lei, N. Morgan, D. Vergyri: Cross-domain and Cross-language Portability of Acoustic Features Estimated by Multilayer Perceptron. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 1, pp. 321–324, Toulouse, France, May 2006.

[Sundermeyer & Nußbaum-Thom+ 11] M. Sundermeyer, M. Nußbaum-Thom, S. Wiesler, C. Plahl, A. El-Desoky Mousa, S. Hahn, D. Nolden, R. Schlüter, H. Ney: The RWTH 2010 Quaero ASR Evaluation System for English, French, and German. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 2212–2215, Prague, Czech Republic, May 2011.

[Sundermeyer & Schlüter+ 12] M. Sundermeyer, R. Schlüter, H. Ney: LSTM Neural Networks for Language Modeling. In *Interspeech*, Portland, OR, USA, Sept. 2012.

[Tóth & Frankel+ 08] L. Tóth, J. Frankel, G. Gasztolya, S. King: Cross-lingual Portability of MLP-based Tandem Features – A Case Study for English and Hungarian. In *Interspeech*, pp. 2695–2698, Brisbane, Australia, Aug. 2008.

[Tüske & Sundermeyer+ 12] Z. Tüske, M. Sundermeyer, R. Schlüter, H. Ney: Context-Dependent MLPs for LVCSR: TANDEM, Hybrid or Both? In *Interspeech*, Portland, OR, USA, Sept. 2012.

[Valente & Hermansky 07] F. Valente, H. Hermansky: Combination of Acoustic Classifiers based on Dempster-Shafer Theory of Evidence. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 1129–1132, Honolulu, HI, USA, April 2007.

[Valente & Magimai-Doss+ 09] F. Valente, M. Magimai-Doss, C. Plahl, S. Ravuri: Hierarchical Processing of the Modulation Spectrum for GALE Mandarin LVCSR System. In *Interspeech*, pp. 2963–2966, Brighton, UK, Sept. 2009.

[Valente & Magimai-Doss+ 11] F. Valente, M. Magimai-Doss, C. Plahl, S. Ravuri, W. Wang: Transcribing Mandarin Broadcast Speech Using Multi-Layer Perceptron Acoustic Features. *IEEE Transactions on Audio, Speech and Language Processing*, Vol. 19, No. 8, Nov. 2011.

[Valente & Vepa+ 07] F. Valente, J. Vepa, C. Plahl, C. Gollan, H. Hermansky, R. Schlüter: Hierarchical Neural Networks Feature Extraction for LVCSR system. In *Interspeech*, pp. 42–45, Antwerp, Belgium, Aug. 2007.

[Vintsyuk 71] T.K. Vintsyuk: Elementwise Recognition of Continuous Speech Composed of Words from a Specified Dictionary. *Kibernetika*, Vol. 7, pp. 133–143, March 1971.

[Viterbi 67] A. Viterbi: Error Bounds for Convolutional Codes and an Asymptotically Optimal Decoding Algorithm. *IEEE Transactions on Information Theory*, Vol. 13, pp. 260–269, 1967.

[von Agris & Kraiss 07] U. von Agris, K.F. Kraiss: Towards a Video Corpus for Signer-Independent Continuous Sign Language Recognition. In *International Workshop on Gesture in Human-Computer Interaction and Simulation*, Lisbon, Portugal, May 2007.

[Vu & Metze$^+$ 12] N.T. Vu, F. Metze, T. Schultz: Multilingual Bottle-neck Features and its Application for Under-resourced Languages. In *International Workshop on Spoken Languages Technologies for Under-resourced Languages (SLTU)*, Cape Town, South Africa, May 2012.

[Waibel & Hanazawa$^+$ 89] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, K.J. Lang: Phoneme Recognition Using Time-delay Neural Networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. 37, No. 3, March 1989.

[Wegmann & McAllaster$^+$ 96] S. Wegmann, D. McAllaster, J. Orloff, B. Peskin: Speaker Normalization on Conversational Telephone Speech. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 339–341, Atlanta, GA, USA, May 1996.

[Werbos 90] P. Werbos: Back Propagation Through Time: What It Does and How to Do It. *Proceedings of the IEEE, Special issue on neural networks*, Vol. 78, pp. 1550–1560, 1990.

[Willett & He 08] D. Willett, C. He: Discriminative training for complementariness in system combination. In *Interspeech*, 919, Brisbane, Australia, Aug. 2008.

[Williams & Zipser 89] R.J. Williams, D. Zipser: A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, Vol. 1, No. 2, pp. 270–280, June 1989.

[Wöllmer & Schuller$^+$ 11] M. Wöllmer, B. Schuller, G. Rigoll: A novel bottleneck-BLSTM front-end for feature-level context modeling in conversational speech recognition. In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 36–41, Hawaii, USA, Dec. 2011.

[Zahedi & Keysers$^+$ 05a] M. Zahedi, D. Keysers, H. Ney: Appearance-Based Recognition of Words in American Sign Language. In *Iberian Conference on Pattern Recognition and Image Analysis*, pp. 513–520, Estoril, Portugal, June 2005.

[Zahedi & Keysers$^+$ 05b] M. Zahedi, D. Keysers, H. Ney: Pronunciation Clustering and Modeling of Variability for Appearance-Based Sign Language Recognition. In *International Workshop on Gesture in Human-Computer Interaction and Simulation*, Vol. 3881, pp. 68–79, Ile-de-Berder, France, May 2005.

[Zell 94] A. Zell: *Simulation Neuronaler Netze*. Addison-Wesley, 1994.

[Zolnay 06] A. Zolnay: *Acoustic Feature Combination for Speech Recognition*. Ph.D. thesis, RWTH Aachen University, Aachen, Germany, Aug. 2006.

[Zolnay & Schlüter$^+$ 05] A. Zolnay, R. Schlüter, H. Ney: Acoustic Feature Combination for Robust Speech Recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 1, pp. 457–460, Philadelphia, PA, March 2005.