Language Model Integration Into Acoustic Model Training

Von der Fakultät für Informatik der RWTH Aachen University
zur Erlangung des akademischen Grades

eines Doktors der Naturwissenschaften

genehmigte Dissertation

vorgelegt von

Wilfried Gero Michel, Master of Science

Berichter:

Universitätsprofessor Dr.-Ing. Hermann Ney

Universitätsprofessor Dr.-Ing. Reinhold Häb-Umbach

Tag der mündlichen Prüfung: 12. Juni 2026

Acknowledgments

I would first like to thank my supervisor Prof. Dr.-Ing. Hermann Ney for giving me the opportunity to join his research group. Throughout my time at the chair he was instrumental in shaping my scientific mindset. He has taught me to work independently, to view the problems from a different angle, and to ask the right questions so that today I find myself asking the very same questions to my students.

I would also like to thank Prof. Dr. Reinhold Häb-Umbach for agreeing to serve as the second supervisor of this thesis and for his interest in my work.

I am likewise grateful to PD Dr. Ralf Schlüter for many stimulating discussions over the years, for his perspective on scientific questions, and for his presence and support throughout my time in the group.

I would like to thank all my colleagues at the Chair of Computer Science VI. Special thanks goes to Pavel, Zoltán, and Mirko who have shown me the ropes of automatic speech recognition and our software. And then, in no further particular order, my thanks to Albert, Andreas, Basha, Benedikt, Christian, Christoph, David, Eugen, Harald, Ilia, Jan, Jan, Julian, Kazuki, Markus, Mohammad, Nick, Oscar, Parnia, Patrick, Peter, Simon, Simon, Tamer, Tina, Tobias, Wei, Weiyue, Yingbo, Yunsu, and Zijian for their collaboration, scientific discussions, helpful implementation details, water cooler chat, emotional support, introducing me to senior colleagues at conferences, and/or sharing their story over a beer.

My thanks also to all of the student workers, who fuel the engine that produces the science that we use in our work. Out of those, I thank in particular Alexander, Daniel, Felix, Minh-Nghia, and Nils-Phillip, who have contributed to this thesis.

Without the support of our admins Stefan and Thomas, not a single number could have been computed, and without the organizational support of Steffi, not a single page of this thesis would have reached the printer. My sincerest thanks go to the people behind the curtain.

And last, but opposite of least, I would like to thank my family. For from early on nurturing my curiosity and interest in natural sciences, for giving me the opportunity and encouragement to pursue my studies, for their support during my time in Aachen, and for taking over extra family duties providing me with the time and space needed to finish up writing down this thesis. Thank you, I will never forget this.

If I have seen further, it is by standing on the shoulders of giants.

ISAAC NEWTON

Abstract

To fuel the training of ever more powerful and accurate speech recognition systems a mountain of bimodal training data is necessary, that includes both the acoustic representation of speech along with the textual transcriptions. The amount of available high-quality training data and the cost of producing more of it is currently the main bottleneck in the development of automatic speech recognition systems. Another component in the improvement of automatic speech recognition systems are language models that capture semantic knowledge of a specific language. These require only textual data and can and have been trained on incomprehensible amounts of it. In this thesis we will investigate how language models can be more tightly integrated into the acoustic model training process to improve the overall system performance.

Firstly, we consider sequence discriminative training of hybrid acoustic models. We propose a novel sequence discriminative training criterion that we call frame-level maximum mutual information. This is closely related to the state-level Bayes risk criterion but using it in model training should move them towards the true distribution instead of a sharp 0-1-distribution. While we do not see any improvements in word error rate, we do observe a qualitative change in the resulting probability distribution, which is less sharp if trained with the frame-level maximum mutual information criterion.

Secondly, we propose to integrate language models into the training process of attention-based encoder-decoder acoustic models via log-linear combination. The sentence-level normalization of the combination gives rise to a sequence discriminative training criterion that we call maximum mutual information, based on its similarity to the equivalent training criterion for hybrid models. We investigate suitable approximations to the normalization term and the required search space size to capture all relevant contributions. The language model used in this combination is tuned extensively to show which properties are optimal for training integration. The new criterion is then compared to the existing expected error criterion and we show that competitive word error rates can be achieved.

Thirdly, we aim to ameliorate the increased training time and complexity of the maximum mutual information criterion by using a symbol-level normalization. We show that the same improvement in word error rates can be achieved without significantly increasing the training time over the baseline. Additionally, we show that symbol-dependent scaling exponents, which are learned by back-propagation, can improve the word error rate. These improvements, however, go away once the acoustic model is jointly trained with the scales and language model.

Lastly, we investigate the interplay of language model integration in acoustic model training and the internal language model correction approach. We find that most of the improvements of integrating a language model in the training process come from suppressing the implicit language model that is learned by the acoustic model decoder. We therefore show that the word error rate improvements obtained by integrating a language model in training are largely already captured when an internal language model correction model is used during recognition.

Kurzfassung

Um das Training immer größerer Spracherkennungssysteme zu befeuern ist ein wachsender Berg an bimodalen Trainingsdaten vonnöten, die sowohl einen Text als auch dessen akustische Repräsentation beinhalten. Die Menge der verfügbaren, qualitativ hochwertigen Trainingsdaten ist aktuell der größte Engpass bei der Weiterentwicklung der automatischen Spracherkennungssysteme. Ein weiterer Ansatz zur Verbesserung der Spracherkennung sind Sprachmodelle, die den semantischen Kontext der Sprache einfangen. Diese benötigen ausschließlich Textdaten und können deshalb - und wurden bereits - auf unvorstellbar großen Datenmengen trainiert. In dieser Arbeit werden wir untersuchen, wie solche Sprachmodelle enger in das Training von akustischen Modellen integriert werden können, um die Qualität des Gesamtsystems zu verbessern.

Zunächst betrachten wir das diskriminative Training von hybriden akustischen Modellen. Wir stellen ein neues diskriminatives Trainingskriterium vor, welches wir frame-level maximum mutual information nennen. Es ist eng mit dem state-level Bayes risk Kriterium verwandt, jedoch sollte dieses im Training angewandt zu einer näher an der wahren Verteilung liegenden Modellverteilung führen. Obwohl wir keine Verbesserung der Wortfehlerrate festgestellt haben, konnten wir doch eine qualitative Veränderung der Modellverteilungen beobachten, welche beim Training mit dem neuen Kriterium deutlich weicher waren.

Anschließend schlagen wir vor, ein Sprachmodell mit Hilfe von log-linearer Kombination in das Training von Attention-basierten Encoder-Decoder Modellen zu integrieren. Die Normierung auf Satzebene führt zu einem diskriminativen Trainingskriterium, welches wir, wegen seiner Ähnlichkeit zum entsprechenden Kriterium für hybride Modelle, maximum mutual information nennen. Wir untersuchen mögliche Näherungen für die Normierung und die nötige Größe des Suchraums, um alle relevanten Beiträge einzufangen. Die Art des zu kombinierenden Sprachmodells wird untersucht um herauszufinden, welche Eigenschaften für eine Trainingsintegration am besten geeignet sind. Das neue Kriterium wird darauf mit dem existierenden expected error Kriterium verglichen und wir erreichen eine konkurrenzfähige Wortfehlerrate.

Daraufhin wollen wir die erhöhte Trainingsdauer und -komplexität des Kriteriums verbessern, indem wir die Normierung stattdessen auf Symbolebene durchführen. Wir zeigen dass die gleichen Verbesserungen in der Wortfehlerrate erreichbar sind ohne jedoch die Trainingsdauer des Referenztrainings signifikant zu erhöhen. Zusätzlich zeigen wir, dass der Modellkombinationsansatz mittels gelernter, symbolabhängiger Skalierungsexponenten verbessert werden kann, solange das akustische Modell nicht gemeinsam mit den Exponenten und dem Sprachmodell trainiert wird.

Zuletzt untersuchen wir den Zusammenhang zwischen einer Sprachmodellintegration im Training und dem internen Sprachmodell, welches implizit im Decoder des akustischen Modells gelernt wird. Wir zeigen, dass der Großteil der Verbesserung durch Trainingsintegration durch ein Unterdrücken des internen Sprachmodells erreicht wird. Deshalb werden die Verbesserungen der Wortfehlerrate durch die Integration eines Sprachmodells in den Trainingsprozess ebenfalls durch ein Modell zur Korrektur des internen Sprachmodells während der Erkennung eingefangen.

Contents

1 Introduction **1**
- 1.1 Automatic Speech Recognition . . . 1
  - 1.1.1 Hybrid Hidden Markov Model Approach . . . 1
  - 1.1.2 Attention-based Encoder-Decoder Approach . . . 2
- 1.2 Knowledge Sources for Automatic Speech Recognition . . . 2
- 1.3 Quality Measures for Automatic Speech Recognition . . . 3
- 1.4 Notation Conventions . . . 4
- 1.5 Software . . . 5

2 Scientific Goals **7**

3 Sequence Discriminative Training of Hybrid Acoustic Models **9**
- 3.1 Introduction . . . 9
- 3.2 Training Criteria . . . 9
  - 3.2.1 Frame Discriminative Cross-Entropy Training Criterion . . . 9
  - 3.2.2 Existing Sequence Discriminative Training Criteria . . . 10
  - 3.2.3 Frame-Level Maximum Mutual Information . . . 12
- 3.3 Experimental Setup . . . 13
  - 3.3.1 Modeling Approach . . . 13
  - 3.3.2 Lattice-based Implementation Details . . . 14
  - 3.3.3 Training Procedure . . . 14
  - 3.3.4 Decision Rule . . . 15
- 3.4 Results . . . 15
  - 3.4.1 Shape of the Probability Distribution . . . 15
  - 3.4.2 Generalized Training Criterion . . . 16
  - 3.4.3 Comparison of Training Criteria . . . 17
  - 3.4.4 Comparison to Other Results in the Literature . . . 17
- 3.5 Conclusions . . . 18
- 3.6 Joint Work and Prior Publications . . . 19

4 Sentence-Level Language Model Integration for Attention-Based Acoustic Models **23**
- 4.1 Introduction . . . 23
- 4.2 Sentence-Level Model Combination Approach . . . 23
- 4.3 Training Criteria . . . 24
  - 4.3.1 Symbol Discriminative Cross-Entropy Training Criterion . . . 24
  - 4.3.2 Sequence Discriminative Training Criteria . . . 24
  - 4.3.3 Denominator Approximation . . . 25

4.4 Experimental Setup . . . 27
4.4.1 Modeling Approach . . . 27
4.4.2 Training Procedure . . . 28
4.4.3 Decision Rule . . . 28
4.5 Results . . . 29
4.5.1 Search Space Size and Approximation . . . 29
4.5.2 Language Model Selection . . . 32
4.5.3 Training Criteria . . . 35
4.5.4 Comparison to Other Results in the Literature . . . 35
4.6 Conclusions . . . 36
4.7 Joint Work and Prior Publications . . . 38

5 Symbol-Level Language Model Integration for Attention-Based Acoustic Models 41
5.1 Introduction . . . 41
5.2 Symbol-Level Model Combination Approach . . . 41
5.3 Training Criteria . . . 42
5.4 Experimental Setup . . . 43
5.4.1 Modeling Approach . . . 43
5.4.2 Training Procedure . . . 43
5.4.3 Decision Rules . . . 43
5.5 Results . . . 44
5.5.1 Automatic Learning of Model Scales . . . 44
5.5.2 Symbol-Dependent Model Scales . . . 45
5.5.3 Decision Rule . . . 46
5.5.4 Language Model Exchange . . . 47
5.5.5 Comparison to Sequence Discriminative Training . . . 48
5.5.6 Comparison to Other Results in the Literature . . . 49
5.6 Conclusions . . . 50
5.7 Joint Work and Prior Publications . . . 53

6 Internal Language Model Correction for Attention-Based Acoustic Models 57
6.1 Introduction . . . 57
6.2 Internal Language Model Estimation . . . 58
6.3 Training Criteria . . . 59
6.4 Experimental Setup . . . 59
6.4.1 Modeling Approach . . . 59
6.4.2 Training Procedure . . . 60
6.4.3 Decision Rule . . . 60
6.5 Results . . . 60
6.5.1 Comparison of Internal Language Model Estimation Methods . . . 60
6.5.2 Internal Language Model Suppression through Training . . . 61
6.5.3 Combining Internal Language Model Suppression and Correction . . . 62
6.5.4 Comparison to Other Results in the Literature . . . 63
6.6 Conclusions . . . 64
6.7 Joint Work and Prior Publications . . . 66

7 Scientific Achievements 69

A Training Corpora 71
A.1 LibriSpeech . . . 71

A.2 Switchboard 71
A.3 Quaero 72

B Derivations **73**
B.1 Gradient of Maximum Mutual Information Criterion 73
B.2 Gradient of State-Level Bayes Risk Criterion 74

C Algorithms **77**
C.1 TensorFlow Implementation of Approximative Recombination 77

List of Figures **83**

List of Tables **85**

Bibliography **87**

1. Introduction

1.1 Automatic Speech Recognition

Automatic speech recognition is the task of automatically mapping a recorded sequence of spoken words to the textual representation of those words. While researchers in the past have tried to explicitly define and write down mapping rules to find such a mapping, it has become clear that data-driven approaches lead in terms of accuracy as well as speed and simplicity [Miclet & Mercier 88]. The current state-of-the-art in automatic speech recognition is to use a statistical approach to assign a probability to each word-sequence hypothesis and then use a decision rule to select a single hypothesis as the output of the system. The word sequence is then broken down into individual modeling units, which can be characters, phonemes, or even full words.

The main task has now become finding the optimal probability distribution $p\left(w_1^N \middle| x_1^T\right)$ to accurately represent human speech. One of the main difficulties in modeling this probability distribution is that the sequence of words or symbols w_1^N and the sequence of acoustic observations x_1^T are not aligned. In this thesis two approaches to creating such an alignment are considered.

1.1.1 Hybrid Hidden Markov Model Approach

The hybrid hidden Markov model approach uses Bayes theorem [Bayes 63] to separate the sentence posterior probability into an acoustic model (AM) that maps the word sequence to its acoustic realization and a language model (LM) that exclusively models the sequence of words.

$$p\left(w_1^N \middle| x_1^T\right) = \frac{p_{AM}\left(x_1^T \middle| w_1^N\right) \cdot p_{LM}\left(w_1^N\right)}{p\left(x_1^T\right)}$$

In the acoustic model a time-synchronous (hidden) state sequence (s_1^T) is then introduced and it is further split into an emission (EM) and a transition model (TM) [Baum & Petrie 66] using a first-order Markov assumption.

$$\begin{aligned} p_{AM}\left(x_1^T \middle| w_1^N\right) &= \sum_{\{s_1^T \in w_1^N\}} p\left(x_1^T, s_1^T \middle| w_1^N\right) \\ &= \sum_{\{s_1^T \in w_1^N\}} \prod_{t=1}^{T} p_{EM}(x_t|s_t) \cdot p_{TM}(s_t|s_{t-1}) \end{aligned}$$

The emission model is then again converted into a discriminative model that is then modeled using neural networks (θ) using Bayes theorem and the state prior model (PM).

$$p_{EM}(x_t|s_t) = \frac{p_\theta(s_t|x_t) \cdot p(x_t)}{p_{PM}(s_t)}$$

In the hybrid approach the sentence posterior is now modeled by a combination of four different models that each cover a different aspect of speech recognition: language model, state transition model, neural emission model, and prior model.

1.1.2 Attention-based Encoder-Decoder Approach

The attention-based encoder-decoder approach aims to directly model the sentence posterior probability using a neural network. The neural network is split into one part (encoder) that runs synchronously with the input time dimension and a second part (decoder) that runs synchronously with the output word position dimension. The alignment problem is explicitly solved by the attention mechanism [Vaswani & Shazeer$^+$ 17], where the input to the decoder is a weighted sum over the encoder states. The weights are determined on the fly based on the past weights and word history. The attention approach can therefore model the full sentence posterior using a single neural network model,

$$p\left(w_1^N \middle| x_1^T\right) = p_{AM}\left(w_1^N \middle| x_1^T\right)$$

which is the reason why these models are sometimes called "end-to-end" models [Prabhavalkar & Hori$^+$ 24].

But for attention-based encoder-decoder models it has also been shown that combining multiple models significantly increases the accuracy of the speech recognition system [Toshniwal & Kannan$^+$ 18]. In state-of-the-art speech recognition systems, the acoustic model is therefore combined with a language model using log-linear model combination.

$$p\left(w_1^N \middle| x_1^T\right) = \frac{1}{Z}\, p_{AM}\left(w_1^N \middle| x_1^T\right) \cdot p_{LM}\left(w_1^N\right)$$

This looks similar to the Bayes decomposition in subsection 1.1.1, but here the acoustic model probability stays a word posterior probability, so that the attention-based encoder-decoder modeling approach can still be used.

1.2 Knowledge Sources for Automatic Speech Recognition

The statistical approach to automatic speech recognition takes advantage of multiple knowledge sources during recognition. These correspond to the different models used in section 1.1. Some knowledge sources stem from human intuition and are hard-coded into the models. These models may consist of HMM topology, a transition model and the pronunciation model, i.e. the lexicon. There are works that try to estimate those models from data, e.g. learning transition models in an expectation-maximization style [Mann & Raissi$^+$ 23], or automatic (re-)estimation of pronunciations [Goronzy & Rapp$^+$ 04], but those remain the exception and no clear and generalizable improvements over hand-crafted models have been found. This is of course with the exception of so-called end-to-end models [Vaswani & Shazeer$^+$ 17], where all of the above models are integrated in the acoustic model and jointly learned.

Other models cannot be efficiently hand-crafted by humans and are instead always estimated on data. These models include the emission model, language model, and prior model. These models have different requirements on the training data: Emission models need matched pairs of audio and transcription, while language models can be estimated solely on standalone text. This leads to an imbalance in the amount of available training data, where acoustic model training data is available on the order of 10^6 to 10^8 words [Godfrey & Holliman$^+$ 92, Panayotov & Chen$^+$ 15], while language model training data is used in the order of 10^{12} words [Touvron & Lavril$^+$ 23, Anil & Dai$^+$ 23], which is caused by a limitation of training resources rather than one of data availability.

Combining knowledge sources is an integral part during the evaluation of statistical speech recognition. But the training of individual models can also take advantage by taking other knowledge sources into account.

- **Increase data seen in training**: When integrating language models into the training process of acoustic models, the training does not rely only on the limited matched data but may also profit from knowledge gained from the large amount of text-only data.
- **Reduce mismatch to recognition**: During recognition the additional knowledge sources will be present. Adding them in the training process reduces the training-recognition mismatch and may direct the acoustic model away from focusing on distinguishing acoustically close but semantically different words and push it to improve in areas where both the acoustic and language model struggle.
- **Optimized error measure**: The training criterion of standalone model training will usually optimize the shape of the model probability distribution to match the training data distribution. This is most often different from the error measure of the task and might lead to degraded accuracy. Including all knowledge sources allows us to use a training criterion that is closer to the error measure of the task.

1.3 Quality Measures for Automatic Speech Recognition

To quantitatively evaluate the quality of automatic speech recognition and to measure the improvement of our methods we need to define quality measures for our speech recognition models.

Word Error Rate (WER)

The *word error rate* is the main error measure used for automatic speech recognition. It is defined as the Levenshtein distance between a hypothesis and the reference transcript. This is computed as the minimum number of edit operations (insertions I, deletions D, substitutions S) needed to transform one sequence into the other, divided by the length N of the reference word sequence w_1^N.

$$\text{WER} := \frac{\text{Levenshtein}(w_1^N, v_1^M)}{N} = \frac{S + I + D}{N}$$

The word error rate is lower bounded by a value of 0 (perfect match between reference and hypothesis) but can be arbitrarily large in the case of many insertions. Lower values are desired.

Additionally, we can define the *word substitution rate* (WSR) where the Levenshtein distance is replaced by the Hamming distance. Here insertions and deletions are not part of the allowed edit operations but only substitutions at the individual word position are allowed. This is sub-optimal as e.g. an otherwise correct sentence missing only the first word gets the same error measure as a completely wrong or an empty sentence. However, this approach is computationally less expensive compared to the Levenshtein distance and may be preferred in certain applications.

Phoneme/Character/Subword Error Rate (PER/CER/SER)

The *phoneme*, *character* or *subword error rate* works in a similar fashion to the word error rate, except in this case the words are decomposed into subunits which are scored individually with Levenshtein or Hamming distance.

In morphologically rich languages this has the advantage that missing the correct ending but getting the word stem correctly results in a lower error count than misrecognizing the entire word. A disadvantage is that longer words, which are generally easier to recognize correctly, disproportionately influence the measure.

Perplexity (PPL)

Perplexity is a quality measure for language models. It is defined as the inverse geometric mean of the word probabilities along a word sequence

$$\mathrm{PPL} = \left[\prod_{n=1}^{N} p\left(w_n \middle| w_0^{n-1}\right)\right]^{-\frac{1}{N}}$$

An intuitive interpretation of the perplexity is that it describes how many plausible words the language model has to distinguish between at each position on average. Lower values are better and the perplexity has a lower bound of 1.

Time Aligned Substitution Rate

An alternative way to count errors is to create a frame-wise time alignment of the word (or subunit) sequences for reference and hypothesis to the audio. Then the Hamming distance between two aligned sequences is taken as an error measure.

Real-Time Factor (RTF)

The *real-time factor* measures the speed with which a certain result can be achieved. It measures the time to complete all computations and divides it by the time of the processed audio. Smaller values are preferable. In parallel applications, one usually also scales with the number of compute units (processor cores or graphics cards).

This measure does not only depend on the algorithms and data used but also on the specific implementation and hardware resources available. Therefore, the absolute numbers cannot serve as a direct comparison across systems. However, for practical applications of automatic speech recognition this plays a crucial role as it directly determines the cost to run the speech recognition application and the time required until results of the experiments are available to the researcher.

1.4 Notation Conventions

In this thesis, we use a particular set of notation which we present in this section. Sequence modeling is the main focus of automatic speech recognition and this thesis. We will represent a sequence of values by a variable with a subscript index denoting the starting index and a superscript index denoting the index of the last item (both inclusive) of a sequence

$$x_a^b := x_a, x_{a+1}, x_{a+2}, \ldots, x_{b-1}, x_b$$

If not noted otherwise, in an iteration (sum, product, ...) over consecutive values a lowercase variable starts at 1 and ends at the value of the uppercase variable:

$$\sum_t x_t := \sum_{t=1}^{T} x_t$$

If not noted otherwise, specific variable names are reserved to have specific meanings:

- x: Acoustic features or a representation thereof
- t, T: Time axis of the acoustic features
- w, v, u: Words or subwords of the output vocabulary. Different letters denote words from a different sequence.

- n, N, m, M, l, L: Position in (lowercase) and length of (uppercase) a word sequence
- a: Output label that models a specific (hidden) state of the speech production
- s, S: Index and length of a sequence of hidden states that model speech production
- θ: Parameters of the acoustic model
- φ: Parameters of the language model

Hybrid Model Conventions

In the hybrid approach to speech recognition the word sequence w_1^N is represented by a sequence of labels a_1^S each of which is modeled by a (hidden) state. For simplicity we will assume that the mapping between word and label sequences is unique. If such a unique mapping does not exist (e.g. pronunciation variants), we will set the probability distribution over the hidden state sequences to uniform so that all terms of $p\left(a_1^S \middle| w_1^N\right)$ vanish.

The term s_1^T denotes an alignment of the state sequence $1, \ldots, S$ to the time sequence $1, \ldots, T$, so that $a_t := a_{s_t}$ is the state label aligned to time t.

Attention Model Conventions

Autoregressive models such as attention-based encoder-decoder models but also language models directly model a sequence of words w_1^N, where the probability of the n-th word depends on all previous words. For $n = 1$ there are no previous words, so the probability of the first word is not explicitly conditional and is only implicitly conditioned on the fact that a sequence starts here.

By introducing a virtual start-of-sentence word $w_0 = <s>$, we make this condition explicit. $p(<s>) = 1$ always holds by definition, therefore $p\left(w_1^N\right) \equiv p\left(w_0^N\right)$. But when handling conditional probabilities we will use the start-of-sentence word to have a well-defined condition.

$$p\left(w_1^N\right) = \prod_n p\left(w_n \middle| w_0^{n-1}\right) = p\left(w_N \middle| w_0^{N-1}\right) \cdot \ldots \cdot p\left(w_2 \middle| w_0^1\right) \cdot p(w_1|w_0) \, [\cdot \, p(w_0) \equiv 1]$$

Similarly, we introduce a virtual end-of-sentence word $</s>$ where $p\left(</s> \middle| w_0^N\right)$ signifies the probability of a sequence ending after the N-th word w_N. For ease of notation we define the sequence length N to include the end-of-sentence word. Therefore w_0^N is a sequence of $N-1$ real words and additionally the virtual start-of-sentence and end-of-sentence words.

1.5 Software

Throughout this thesis we will use RETURNN, the RWTH extensible training framework for universal recurrent neural networks [Doetsch & Zeyer$^+$ 17, Zeyer & Alkhouli$^+$ 18], which is a Python wrapper around Theano [Bergstra & Breuleux$^+$ 10] (now deprecated), TensorFlow [Abadi & Barham$^+$ 16], and PyTorch [Paszke & Gross$^+$ 19] to simplify the design and implementation of neural network models and management of datasets. In this thesis exclusively the Tensorflow part will be used.

Additionally, in chapter 3 RASR, the RWTH Aachen University speech recognition system [Rybach & Gollan$^+$ 09, Wiesler & Richard$^+$ 14] is used to generate the lattices, calculate the sequence discriminative training objective functions, and perform the search during recognition.

As a workflow management system we use Sisyphus [Peter & Beck$^+$ 18] together with the recipe collection based around automatic speech recognition `i6_core`.[1]

[1] Available online at `https://github.com/rwth-i6/i6_core`

2. Scientific Goals

In this thesis we want to investigate how acoustic models can be improved by a tighter integration of language models into the training process. To this end, we follow multiple approaches for different kinds of models.

Sequence Discriminative Training of Hybrid Acoustic Models

For hybrid models, where language model integration via sequence discriminative training is already well-established, we want to investigate how the existing criteria can be improved. By reformulating the state-level Bayes risk criterion to a frame-wise posterior contribution and adding a logarithm similar to the maximum mutual information criterion, we arrive at a novel training criterion that we call the frame-level maximum mutual information criterion.

We want to investigate if this criterion leads to improved word error rates and/or a model distribution that is otherwise more beneficial for downstream tasks.

Sentence-Level Language Model Integration for Attention-Based Acoustic Models

For attention-based encoder-decoder acoustic models there is no well-established method to integrate a language model into the training process. We will investigate how language models can be integrated in the training process using log-linear combination of the sentence-level probabilities.

This form of integration requires a normalization over the set of all possible word sequences, which is intractable to compute. We will investigate two approximation methods for this sum, n-best lists and trellises with approximative recombination, to find the required search space size to capture the necessary number of hypotheses in the normalization.

Also, it is unclear which language model should be chosen for integration into the training process. We will vary language model context, number of parameters, training data, and training time of the language model to find the optimal model for integration.

Symbol-Level Language Model Integration for Attention-Based Acoustic Models

Sentence-level integration leads to an increase in training time and complexity. We aim to find an integration method that gives a similar improvement in word error rate but does not increase the training complexity as much. For this, we will investigate symbol-level language model integration in the training process, to see how much improvement we can get compared to sequence discriminative training.

If this combination approach is also used consistently in recognition, this would incur a degradation in recognition speed. We will investigate whether it is possible to use sentence-level combination during recognition even if symbol-level combination was used in training.

Additionally, we investigate if in this approach the model combination weights can also be learned and if symbol-dependent scaling exponents are helpful.

Language Model Correction for Attention-Based Acoustic Models

For encoder-decoder models, it is possible to implicitly learn an internal language model during training. Correcting for this internal language model can significantly improve the word error rate. We will compare different approaches to estimating this internal language model to find which captures it most accurately.

We will also investigate the interplay of language model integration in training and internal language model correction during recognition. We will investigate whether integrating a language model in training influences the internal language model learned in the decoder and whether both methods can be combined to further improve the word error rate of the final system.

3. Sequence Discriminative Training of Hybrid Acoustic Models

3.1 Introduction

Hybrid acoustic modeling is one of the oldest approaches to including a neural network into the acoustic modeling process [Morgan & Bourlard$^+$ 93]. It stays closest to the original speech recognition framework where a hidden Markov model is used to align the sequence of observations to the spoken words. In hybrid acoustic modeling, the HMM topology and transition probabilities are kept but the emission probabilities are modeled via a deep neural network instead of Gaussian mixture distributions.

Even for the Gaussian mixture models it was common to incorporate language models into the parameter estimation by using a discriminative training criterion, e.g. maximum mutual information (MMI) [Bahl & Brown$^+$ 86], minimum error (ME) [Chou & Lee$^+$ 93], or corrective training [Bahl & Brown$^+$ 88]. These methods have successfully been applied to hybrid acoustic models with feed-forward neural architecture [Veselý & Ghoshal$^+$ 13] and also to recurrent [Voigtlaender & Doetsch$^+$ 15] and attention-based neural models [Lee & Chang 22].

In this chapter, frame-level maximum mutual information, a novel sequence discriminative training criterion, is introduced and compared to the existing training criteria maximum mutual information and state-level Bayes risk. In the following, in section 3.2, first the existing training criteria are presented and then the novel criterion is introduced in subsection 3.2.3. The details of the experimental setup are given in section 3.3, including a description of the neural models and the training procedure that is used. The results are presented in section 3.4.

3.2 Training Criteria

3.2.1 Frame Discriminative Cross-Entropy Training Criterion

All frame-level training criteria assume that a one-to-one mapping of the input features x_1^T to reference labels a_1^T exists. This should be obtained prior to the training, e.g. via alignment with a different, previously trained acoustic model or by manual annotation. Given this mapping called *alignment*, the training criterion is optimized for each time step t independently.

The *cross-entropy* training criterion is the most widely used criterion for neural network models [Huang & Li$^+$ 14, Graves & Jaitly$^+$ 13]. It is defined as the cross-entropy of the normalized model output distribution $p_\theta(a|x_1^T)$ and the empirical distribution of the training data $\mathcal{T}$

$$F_{\mathrm{CE}}(\theta) := \sum_{(a_1^S, x_1^T) \in \mathcal{T}} \max_{s_1^T \in a_1^S} \sum_t \log\, p_\theta(a_{s_t}|x_1^T) \tag{3.1}$$

where the contributions of all time steps t are summed up to obtain the full training criterion for a sentence or dataset.

To optimize this training criterion, the value that the model assigns to the reference label should be maximized. Due to the normalization constraint over all labels, this implies that the value for all other labels is minimized at the same time. Therefore neural models are trained frame-discriminatively with the cross-entropy criterion.

3.2.2 Existing Sequence Discriminative Training Criteria

Sequence discriminative training criteria, as opposed to frame discriminative criteria, take into account a sequence of inputs and outputs to define the optimization criterion. For sequence modeling tasks such as speech recognition, these are closer to the final evaluation measure.

If the sequence considered expands further than single sounds or words into the magnitude of whole sentences, then these criteria also offer the possibility of incorporating a language model into the training process.

Maximum Mutual Information (MMI)

The maximum mutual information criterion is the log posterior probability of the reference word sequence w_1^N given the audio data x_1^T. This can, in analogy to the frame-wise cross-entropy criterion, be seen as a sentence-level cross-entropy criterion between the model-assigned probability of a sentence $p\left(v_1^M \middle| x_1^T\right)$ and the empirical distribution over the training data $\mathcal{T}$.

$$F_{\text{MMI}} = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \log p\left(w_1^N \middle| x_1^T\right)$$

Then the usual decomposition into acoustic model (AM) and language model (LM) is employed and a sequence of time-aligned states (s_1^T) is introduced as a hidden variable

$$\begin{aligned} &= \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \log \frac{p_{\text{AM}}\left(x_1^T \middle| w_1^N\right) \cdot p_{\text{LM}}\left(w_1^N\right)}{p(x_1^T)} \\ &= \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \log \frac{\sum\limits_{\{s_1^T \in w_1^N\}} p_{\text{AM}}\left(x_1^T, s_1^T \middle| w_1^N\right) p_{\text{LM}}\left(w_1^N\right)}{\sum\limits_{M, \{v_1^M\}} \sum\limits_{\{s_1^T \in v_1^M\}} p_{\text{AM}}(x_1^T, s_1^T | v_1^M)\, p_{\text{LM}}(v_1^M)} \end{aligned} \tag{3.2}$$

After inserting the hybrid model decomposition and introducing scaling exponents, we arrive at the well-known result:

$$F_{\text{MMI}}(\theta) = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \log \frac{p^\beta\left(w_1^N\right) \sum\limits_{\{(a_1^S, s_1^T) \in w_1^N\}} \prod_{t=1}^T p_\theta^\alpha\left(a_{s_t} \middle| x_1^T\right)\ p^{-\gamma}(a_{s_t})\ p^\delta(s_t | s_{t-1})}{\sum\limits_{M, \{v_1^M\}} p^\beta(v_1^M) \sum\limits_{\{(a_1^S, s_1^T) \in v_1^M\}} \prod_{t=1}^T p_\theta^\alpha(a_{s_t} | x_1^T)\ p^{-\gamma}(a_{s_t})\ p^\delta(s_t | s_{t-1})} \tag{3.3}$$

Where $M, \{v_1^M\}$ means to consider the set of all possible word sequences of any length and $\{(a_1^S, s_1^T) \in w_1^N\}$ means the set of all HMM state (s) and label (a) sequences that are compatible with the given word sequence. In practice, the sum over all word sequences is restricted to a lattice $\mathcal{L}$ of relevant hypotheses, and the sum over HMM states is replaced by the maximum, so that we arrive at:

$$F_{\text{MMI}}(\theta) = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \log \frac{p^\beta\left(w_1^N\right) \max\limits_{\{(a_1^S, s_1^T) \in w_1^N\}} \prod_{t=1}^T p_\theta^\alpha\left(a_{s_t} \middle| x_1^T\right)\ p^{-\gamma}(a_{s_t})\ p^\delta(s_t | s_{t-1})}{\sum\limits_{M, \{v_1^M\} \in \mathcal{L}} p^\beta(v_1^M) \max\limits_{\{(a_1^S, s_1^T) \in v_1^M\}} \prod_{t=1}^T p_\theta^\alpha(a_{s_t} | x_1^T)\ p^{-\gamma}(a_{s_t})\ p^\delta(s_t | s_{t-1})} \tag{3.4}$$

As can be best seen in the form of Equation 3.2, the training criterion will be optimized by maximizing the probability of the reference word sequence while minimizing the probability of all other sequences. With this criterion, whole sequences are weighted against each other. Therefore it is considered to be a sequence discriminative criterion.

The gradient of the maximum mutual information criterion is given by the difference

$$\frac{\partial F_{\mathrm{MMI}}}{\partial \log p_\theta(a|x_1^T)} = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \alpha \left[\gamma^N\left(a_{s_t} = a \middle| x_1^T, w_1^N\right) - \gamma^D\left(a_{s_t} = a \middle| x_1^T\right) \right] \tag{3.5}$$

with γ^N and γ^D as derived in Appendix B.1.

The denominator state occupancy $\gamma^D\left(a_{s_t} = a \middle| x_1^T\right)$ is the probability that, given the audio x_1^T and considering all other knowledge sources, at the time t the label is a. Similarly, the numerator state occupancy $\gamma^N\left(a_{s_t} = a \middle| x_1^T, w_1^N\right)$ is the probability that at the time t the label is a, given the audio x_1^T, all other knowledge sources, and with additional knowledge of the reference word sequence w_1^N.

Bayes Risk Related Criteria (sMBR/MPE)

The Bayes risk (MBR) related training criteria are a family of sequence discriminative training criteria that focus on minimizing the expected error with respect to an accuracy measure $A(\cdot, \cdot)$. The training criteria follow the general structure of

$$F_{\mathrm{MBR}} = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \sum_{M, \{v_1^M\}} p\left(v_1^M \middle| x_1^T\right) A\left(v_1^M, w_1^N\right)$$

where the accuracy measure $A\left(v_1^M, w_1^N\right)$ is between a hypothesis word sequence v_1^M and the reference word sequence w_1^N.

Different choices of accuracy measure give rise to specific training criteria. A simple equality check between the two sequences

$$A\left(v_1^M, w_1^N\right) := \delta_{v_1^M, w_1^N}$$

would optimize the sentence error rate and lead to a criterion similar to maximum mutual information - with the exception of a missing logarithm.

Ideally one would choose the accuracy measure that is later applied for scoring the final model, i.e. word error rate or Levenshtein edit distance

$$A\left(v_1^M, w_1^N\right) := -\mathrm{Levensthein}\left(v_1^M, w_1^N\right).$$

This criterion, called *minimum word error rate* (MWE), however, relies on a non-local accuracy measure and is impractical to compute efficiently. As an approximation one could use the Hamming distance instead

$$A\left(v_1^M, w_1^N\right) := |M - N| + \sum_{k=1}^{\min(M,N)} \delta_{v_k, w_k}$$

but this would discount completely different words as much as words that differ by only a single phoneme.

An option to reward partially correct words is to consider the phonetics of both word sequences. Here not two word sequences, but two phoneme sequences are compared. The reference phoneme sequences are first time-aligned by

$$(\omega_1^\Sigma, \sigma_1^T) := \underset{\{(a_1^S, s_1^T) \in w_1^N\}}{\arg\max} \prod_{t=1}^{T} p^\alpha\Big(a_{s_t}\Big|x_1^T\Big)\ p^{-\gamma}(a_{s_t})\ p^\delta(s_t|s_{t-1})$$

and then the Hamming distance between the currently hypothesized phoneme sequences $a_{s_1}^{s_T}$ and the reference alignment is used as accuracy measure

$$A\Big(v_1^M, w_1^N\Big) := \sum_{t=1}^{T} \delta_{a_{s_t}, \omega_{\sigma_t}}.$$

If only the (phoneme) labels (a) are compared, then the criterion is called *minimum phoneme error rate* (MPE) [Povey & Woodland 02]. If labels (a, ω) and HMM-states (s, σ) are compared, then the criterion is called *state-level Bayes risk* (sMBR) [Gibson & Hain 06].

Using the same expansions as in Equation 3.4, we can write the state-level Bayes risk criterion in expanded form:

$$F_{\text{sMBR}}(\theta) = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \frac{\sum\limits_{M, \{v_1^M\} \in \mathcal{L}} p^\beta\Big(v_1^M\Big) \max\limits_{\{(a_1^S, s_1^T) \in v_1^M\}} \prod\limits_{t=1}^{T} p_\theta^\alpha\Big(a_{s_t}\Big|x_1^T\Big)\ p^{-\gamma}(a_{s_t})\ p^\delta(s_t|s_{t-1}) \cdot \sum\limits_{\tau=1}^{T} \delta_{a_{s_\tau}, \omega_{\sigma_\tau}}}{\sum\limits_{M, \{v_1^M\} \in \mathcal{L}} p^\beta(v_1^M) \max\limits_{\{(a_1^S, s_1^T) \in v_1^M\}} \prod\limits_{t=1}^{T} p_\theta^\alpha(a_{s_t}|x_1^T)\ p^{-\gamma}(a_{s_t})\ p^\delta(s_t|s_{t-1})}$$

This can be reordered and simplified to the denominator state occupancy (see Appendix B.1).

$$F_{\text{sMBR}} = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \sum_{t=1}^{T} \gamma^D\Big(a_{s_t} = \omega_t\Big|x_1^T\Big) \tag{3.6}$$

In this form, the state-level Bayes risk criterion is also very similar to a cross-entropy criterion in the form of Equation 3.1. Only the probability in question is not the neural model output probability but the state occupancy, and the logarithm is not applied.

The gradient of the state-level Bayes risk criterion can be written in the form of

$$\frac{\partial F_{\text{sMBR}}}{\partial \log p_\theta(a|x_1^T)} = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \alpha \gamma^D\Big(a_{s_t} = a\Big|x_1^T\Big) \Big[\gamma^A - \bar{A}\Big]$$

as derived in Appendix B.2.

3.2.3 Frame-Level Maximum Mutual Information

In the asymptotic case of infinite modeling power, the probability model that optimizes the cross-entropy training criterion will approach the empirical distribution. The minimum error-based criteria based on a zero-one-loss are optimized by the mode of the empirical distribution, i.e. a distribution that is 1 for the most likely candidate and 0 otherwise [Schlüter & Ney 01].

In the spirit of making the state-level Bayes risk criterion more closely resemble a cross-entropy we define a novel training criterion, which we denote *frame-level maximum mutual information* (F-MMI) criterion. For this, we add a logarithm to the state-level Bayes risk criterion of Equation 3.6.

$$F_{\text{F-MMI}} := \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \sum_{t=1}^{T} \log\ \gamma^D\Big(a_{s_t} = \omega_{\sigma_t}\Big|x_1^T\Big)$$

where γ^D is the marginalized sequence posterior probability of the label a at time t and the reference label ω_{σ_t} as defined in subsubsection 3.2.2. In the expanded form the equation reads

$$F_{\text{F-MMI}}(\theta) = \sum_{(w_1^N, x_1^T)\in\mathcal{T}} \sum_{t=1}^{T} \log \frac{\sum\limits_{M,\{v_1^M\}\in\mathcal{L}} p^{\beta}\left(v_1^M\right) \max\limits_{\substack{\{(a_1^S, s_1^T)\in v_1^M\} \\ a_{s_t}=\omega_{\sigma_t}}} \prod_{\tau=1}^{T} p_\theta^{\alpha}\left(a_{s_\tau}\middle|x_1^T\right)\ p^{-\gamma}(a_{s_\tau})\ p^{\delta}(s_\tau|s_{\tau-1})}{\sum\limits_{M,\{v_1^M\}\in\mathcal{L}} p^{\beta}\left(v_1^M\right) \max\limits_{\{(a_1^S, s_1^T)\in v_1^M\}} \prod_{\tau=1}^{T} p_\theta^{\alpha}\left(a_{s_\tau}|x_1^T\right)\ p^{-\gamma}(a_{s_\tau})\ p^{\delta}(s_\tau|s_{\tau-1})} \tag{3.7}$$

For the gradient of the frame-level maximum mutual information criterion

$$\begin{aligned}
\frac{\partial}{\partial\theta} F_{\text{F-MMI}} &= \sum_{(w_1^N, x_1^T)\in\mathcal{T}} \sum_{t=1}^{T} \frac{\partial}{\partial\theta} \log \gamma^D\left(a_{s_t} = \omega_{\sigma_t}\middle|x_1^T\right) \\
&= \sum_{(w_1^N, x_1^T)\in\mathcal{T}} \sum_{t=1}^{T} \frac{1}{\gamma^D\left(a_{s_t} = \omega_{\sigma_t}|x_1^T\right)} \frac{\partial}{\partial\theta} \gamma^D\left(a_{s_t} = \omega_{\sigma_t}\middle|x_1^T\right) \\
&= \sum_{(w_1^N, x_1^T)\in\mathcal{T}} \sum_{t=1}^{T} \frac{1}{\gamma^D\left(a_{s_t} = \omega_{\sigma_t}|x_1^T\right)} \frac{\partial}{\partial\theta} F_{\text{sMBR}}(t)
\end{aligned} \tag{3.8}$$

we can reuse the state-level Bayes risk gradient computation with an additional factor of $\frac{1}{\gamma^D}$.

Generalized Frame-Level Maximum Mutual Information The simple form of the gradient in Equation 3.8 allows for a definition of a generalized criterion that continuously interpolates between the frame-level maximum mutual information and the state-level Bayes risk criterion. For this criterion the pre-factor of the gradient is modified to include a scaling parameter α:

$$\frac{1}{\alpha + (1-\alpha)\,\gamma^D(a_{s_t} = a|x_1^T)}. \tag{3.9}$$

For $\alpha = 0$ we recover the original frame-level maximum mutual information criterion, and in the case of $\alpha = 1$ we arrive at the state-level Bayes risk criterion.

3.3 Experimental Setup

3.3.1 Modeling Approach

In our experiments we use the hybrid DNN-HMM approach with a bidirectional long short-term memory neural network (BLSTM) [Hochreiter & Schmidhuber 97] and a hidden Markov model (HMM) with 0-1-2-topology. The transition probabilities are two hand-crafted sets of values for loop, forward, and skip each for silence and for speech states. The probabilities are given in Table 3.1 and are based on prior findings and are not subject to extensive tuning as they are assumed to have little influence on the recognition performance.

Table 3.1: Default HMM transition probabilities used throughout this chapter.

transition	speech	silence
loop	0.05	0.95
forward	0.95	0.05
skip	10^{-13}	–

The BLSTM emission model consists of 6 layers with 500 units (1000 for LibriSpeech) each for forward and backward directions and an additional linear layer that projects the last BLSTM outputs onto the NN output dimension for each corpus. During training, at each layer's input 10% of the values are randomly selected and set to zero, while the remaining values are multiplied by a factor of 1.1 to compensate for the lost inputs [Srivastava & Hinton$^+$ 14]. Additionally to the training criteria described below, we add the L2 norm of the model parameters with a weight of 10^{-2} to the objective function as a means of regularization. As input features we select 50-dimensional Gammatone features [Schlüter & Bezrukov$^+$ 07] (40-dim for Switchboard) with a frame size of 25ms and a shift of 10ms. These features are normalized per training utterance.[1]

The prior model (PM) is estimated from the neural emission model by averaging the output probabilities over a portion of the training data [Manohar & Povey$^+$ 15].

The language model (LM) used in training is a count-based bigram model that is estimated on the training transcriptions. During recognition a fourgram count-based language model trained on additional unpaired text is used.

3.3.2 Lattice-based Implementation Details

In lattice-based sequence training, the relevant competing hypotheses are obtained and stored prior to the training run, by running speech recognition on the training data. But instead of reporting only the word sequence with maximum probability, a threshold is selected and all sequences which have a probability higher than this threshold are reported. Usually the speech recognition algorithm uses a hypothesis-pruning scheme so that the pruning threshold is chosen accordingly.

If the underlying probabilistic models allow for recombination of hypotheses, this is used to reduce the representation size of the hypothesis space. An efficient representation of the search space is a *lattice*. A lattice is a structure that has a (unique) start state, that represents the start of the utterance. From this start state there are outgoing arcs for each word that is hypothesized. Each arc ends in another state, that represents the ending time of the recognized word and the starting time of the next word. If two hypotheses end at the same time and can be merged according to the underlying probabilistic models, then their arcs end in the same state. Each arc in the lattice represents one recognized word, which is stored with its word-id, word-start and -end times, and the best matching HMM-state sequence that describes that word.

During sequence training, the lattice structure stays the same. Therefore, the language model contributions also do not change and can be kept alongside the lattice. The acoustic model, however, is updated in training, and the acoustic model contributions are updated in every step using the fixed frame-wise alignments. The state occupation probabilities are then calculated using a forward-backward procedure over the lattice and fed back into the training routine of the acoustic model.

3.3.3 Training Procedure

Sequence discriminative training criteria need additional computational resources to be computed and this significantly slows down the training process. Additionally, to generate lattices for lattice-based sequence training, a seed model is needed. To act on these challenges, the following training schedule is used.

As a first step, we train one seed model using only the cross-entropy criterion until convergence. All necessary model scales and the initial learning rate are tuned using this seed model. This seed model is then used for recognition on the training data to generate the lattices. Then the

[1] As an example, the LibriSpeech system definition is available online at `https://github.com/rwth-i6/returnn-experiments/tree/master/2019-librispeech-system/hybrid`

seed model is used as an initialization for a second training step with a sequence discriminative training criterion. In this step the lattice topology and time information is kept constant and only the acoustic scores are updated. To eliminate the effect of the additional training time for the sequence discriminatively trained models, we also continue training with the cross-entropy criterion for the same number of epochs and report this as the "baseline" result.

3.3.4 Decision Rule

During recognition we use the trained models to find the word sequence that has been spoken in an unknown audio sequence. For this we use the maximum a posteriori (MAP) decision rule that picks the word sequence that has the highest probability

$$x_1^T \mapsto w_1^N = \arg\max_{M,\{v_1^M\}} p\Big(v_1^M \Big| x_1^T\Big) \,.$$

after applying the usual hybrid model decomposition, Viterbi approximation, and scaling exponents, the complete decision rule is

$$x_1^T \mapsto w_1^N = \arg\max_{M,\{v_1^M\}} \left[p_{\mathrm{LM}}^{\beta}\Big(v_1^M\Big) \max_{\{(a_1^S, s_1^T) \in v_1^M\}} \prod_{t=1}^{T} p_{\mathrm{EM}}^{\alpha}\Big(a_{s_t}\Big|x_1^T\Big)\; p_{\mathrm{PM}}^{-\gamma}(a_{s_t})\; p_{\mathrm{TM}}^{\delta}(s_t|s_{t-1}) \right] \tag{3.10}$$

with the language model p_{LM}, emission model p_{EM}, prior model p_{PM}, and transition model p_{TM} as defined in subsection 3.3.1.

3.4 Results

3.4.1 Shape of the Probability Distribution

As was discussed in subsection 3.2.3, the frame-level maximum mutual information criterion should push the neural model away from a sharp zero-one distribution towards the true distribution. In this section we want to verify if this can be observed in the probability distributions that are learned with this criterion.

To measure the sharpness of the distributions, we calculate the average entropy S of the probability distribution over the training data $\mathcal{T}$.

$$S := -\frac{1}{Z} \sum_{x_1^T \in \mathcal{T}} \sum_t \sum_a p_t\Big(a\Big|x_1^T\Big) \cdot \log p_t\Big(a\Big|x_1^T\Big) \tag{3.11}$$

Since the model is generally very confident in deciding between speech and non-speech frames, the entropy for silence frames nearly vanishes. Instead we consider only speech frames according to the reference alignment and average the entropy value over the training data that are aligned to speech frames.

The results are presented in Table 3.2. Compared to the probability distribution trained with the state-level Bayes risk criterion, the probability distribution trained with the frame-level maximum mutual information criterion has a much larger entropy. On the Switchboard corpus it increases by 15% and on Quaero by 81%. This indicates an effective broadening of the probability distributions by including the logarithm.

The probability distribution trained with cross-entropy shows very similar values to the probability distribution trained with state-level Bayes risk, lying within 1% of each other.

Table 3.2: Average entropy of the trained neural emission model for the Switchboard (SWB) and Quaero corpus. The average is taken only with respect to speech frames; silence frames according to the reference alignment were excluded.

training criterion	average entropy S	
	SWB	Quaero
cross-entropy	1.08	1.13
state-level Bayes risk	1.09	1.12
frame-level maximum mutual information	1.25	2.03

A similar trend can also be observed when the average probability distribution is plotted. In Figure 3.1 the probability distribution is averaged over the speech frames of the training corpora. The labels are then sorted by descending probability value. Then for each position n the probability mass of the first n labels is accumulated and displayed.

For both corpora, the curves for cross-entropy and state-level Bayes risk lie very close together. This confirms the observation from the entropy comparison. The curve for the frame-level maximum mutual information criterion consistently lies below the other curves, meaning that more classes need to be accumulated to arrive at the same cumulative probability value or that the probability is spread out among more classes.

In the case of the generalized criterion, the parameter α directly controls the sharpness of the distribution. While values smaller than one lead to curves between state-level Bayes risk and frame-level maximum mutual information, values larger than one even further sharpen the probability distributions.

3.4.2 Generalized Training Criterion

In paragraph 3.2.3 a generalized criterion that interpolates between frame-level maximum mutual information and state-level Bayes risk was introduced. The definition in Equation 3.9 contained an interpolation factor α whose influence on the recognition performance is investigated.

In Table 3.3 the word error rate for models trained with different values for the interpolation factor α are given, along with the name of the training criterion if applicable.

Table 3.3: Influence of the interpolation factor α of the generalized training criterion on sequence discriminative training performance compared to the cross-entropy (CE) baseline and the frame-level maximum mutual information (F-MMI) and state-level Bayes risk (sMBR) criteria. Given are the word error rates (WER) on the dev'10 set for Quaero and the Hub5'00 set for Switchboard.

Interpolation factor α	equivalent training criterion name	WER [%]	
		Hub5'00	dev'10
-	CE	14.7	14.0
0.0	frame MMI	14.4	13.0
0.1	-	14.0	12.8
0.5	-	14.0	12.9
1.0	sMBR	14.0	12.8
1.5	-	14.1	12.9
3.0	-	14.1	13.0

For the dev'10 dataset that belongs to the Quaero corpus, no significant change with varying α can be observed. All error rates lie in the range of $12.9\% \pm 0.1\%$. Compared to the cross-entropy baseline of 14.0%, all sequence discriminative criteria lead to a significant word error rate improvement of 8% relative.

For the Hub5'00 dataset that belongs to the Switchboard corpus, the value $\alpha = 0.0$, which is equivalent to the frame-level maximum mutual information criterion, leads to a significantly higher error rate of 14.4% compared to all other values of α, which lie around $14.0\% \pm 0.1\%$. Compared to the cross-entropy baseline of 14.7%, all sequence discriminative criteria improve the word error rate. For frame-level maximum mutual information the relative improvement is 2%, for all other criteria it is 5%.

The special point of $\alpha = 1.0$ that coincides with the state-level Bayes risk criterion is not significantly different from surrounding values.

3.4.3 Comparison of Training Criteria

The recognition performance of the models trained with different training criteria is depicted in Table 3.4. In terms of word error rate it shows improvement for all sequence discriminative criteria compared to the cross-entropy baseline. Between the proposed sequence discriminative criteria there is, however, no significant quality difference that generalizes over all datasets.

Table 3.4: Word error rates (WER) in percent for all tested sequence discriminative training criteria, sentence- and frame-level maximum mutual information (MMI), state-level Bayes risk (sMBR) and the cross-entropy (CE) baseline. All hyperparameters were optimized on Hub5'00 for Switchboard (SWB), and the respective dev sets for Quaero and LibriSpeech (LBS).

training	SWB		Quaero		LBS clean		LBS other	
criterion	Hub5'00	Hub5'01	dev'10	eval'13	dev	test	dev	test
CE	14.7	14.7	14.0	10.4	4.0	4.5	9.5	10.0
sentence MMI	14.0	13.8	13.0	9.8	3.5	3.9	8.4	9.0
sMBR	14.0	13.9	12.8	9.8	3.4	3.7	8.2	8.8
frame MMI	14.4	14.2	13.0	9.7	3.5	3.9	8.5	9.1

On the Switchboard corpus, the proposed frame-level maximum mutual information training criterion underperforms compared to the existing criteria for both dev and test sets. On the Quaero corpus, the word error rates are comparable to the sentence-level maximum mutual information criterion, and for the eval'13 set the frame-level maximum mutual information outperforms all other criteria by a small amount. On LibriSpeech, the frame-level maximum mutual information criterion is on par with the sentence-level maximum mutual information criterion on all sets. Both are outperformed by the state-level Bayes risk criterion by a small amount.

3.4.4 Comparison to Other Results in the Literature

Training Criteria

In this section we compare the results and improvements we observed when using sequence discriminative training criteria to results reported by other groups. Table 3.5 shows an overview of relevant work in the field of sequence discriminative training for hybrid models.

Most authors report relative word error rate improvements in the range of 5% to 10%, which is in line with our findings in this work. If multiple training criteria are compared, then the resulting word error rates seem to vary between the criteria. There is no clear optimal training criterion.

If a group selects a single training criterion for its systems, then most often it is the state-level Bayes risk criterion or, more recently, a lattice-free version of the maximum mutual information criterion LF-MMI, introduced in [Povey & Peddinti$^+$ 16].

3.5 Conclusions

In this chapter a novel sequence discriminative training criterion, the frame-level maximum mutual information, was introduced. This criterion is an extension to the existing state-level Bayes risk criterion which is extended by adding a logarithm. With this extension we aimed to encourage the trained model distribution to more closely follow the true probability distribution. We compared this criterion to the state-level Bayes risk criterion and the standard sentence-level maximum mutual information criterion.

Word Error Rates

We found that the frame-level maximum mutual information training criterion yields comparable word error rates to the existing training criteria. In Table 3.4 we showed that the word error rate on the Quaero eval set was improved from 9.8% for state-level Bayes risk to 9.7% for frame-level maximum mutual information, whereas there was degradation from 8.8% to 9.1% on LibriSpeech test-other and from 13.9% to 14.2% on Switchboard Hub5'01.

We conclude that the performance in terms of word error rate is on par across all sequence discriminative training criteria investigated and see no reason to prefer one criterion over the other.

Shape of Model Distribution

Additionally we have investigated the shape of the resulting probability distribution that arises from the model parameters trained with the different training criteria. In Figure 3.1 we showed that the acoustic models trained with the frame-level maximum mutual information criterion exhibited a smoother probability distribution compared to models trained with the state-level Bayes risk criterion.

On Quaero, the average value of the highest model probability was 47% when the model was trained with the frame-level maximum mutual information criterion and 65% when trained with the state-level Bayes risk criterion. The probability mass covered by the ten highest values was on average reduced from 96% for state-level Bayes risk to 87% for frame-level maximum mutual information.

On Switchboard, we found the same trends albeit to a lesser extent. The average value of the highest model probability was reduced from 69% for state-level Bayes risk to 66% for frame-level maximum mutual information, and the average probability mass covered by the ten highest values was reduced from 97% to 96%.

Verdict

All sequence discriminative training criteria do outperform the frame discriminative cross-entropy criterion by 5%-10% relative improvement. It is therefore confirmed that including additional knowledge sources such as language model information into the acoustic model training process is beneficial.

Although our proposed criterion did not outperform the existing criteria in terms of word error rate, it is our estimation that models trained with the frame-level maximum mutual information training criterion are advantageous for certain downstream tasks such as keyword spotting or compliance that do not only need the single best recognized word sequence but also rely on a rich coverage of the search space.

3.6 Joint Work and Prior Publications

The main findings in this chapter were published in [Michel & Schlüter$^+$ 20b]. For the Switchboard and Quaero datasets, all neural models were trained and evaluated by the author.

For LibriSpeech, the cross-entropy baseline was trained by Christoph Lüscher. Sequence discriminative training and evaluation was done by the author. The results for LibriSpeech were partially published in [Lüscher & Beck$^+$ 19].

The implementation of the sentence-level maximum mutual information and state-level Bayes risk criterion for neural networks was done by Simon Wiesler and Ilia Kulikov based on prior work for Gaussian mixture models. The implementation of the frame-level maximum mutual information criterion was done by the author.

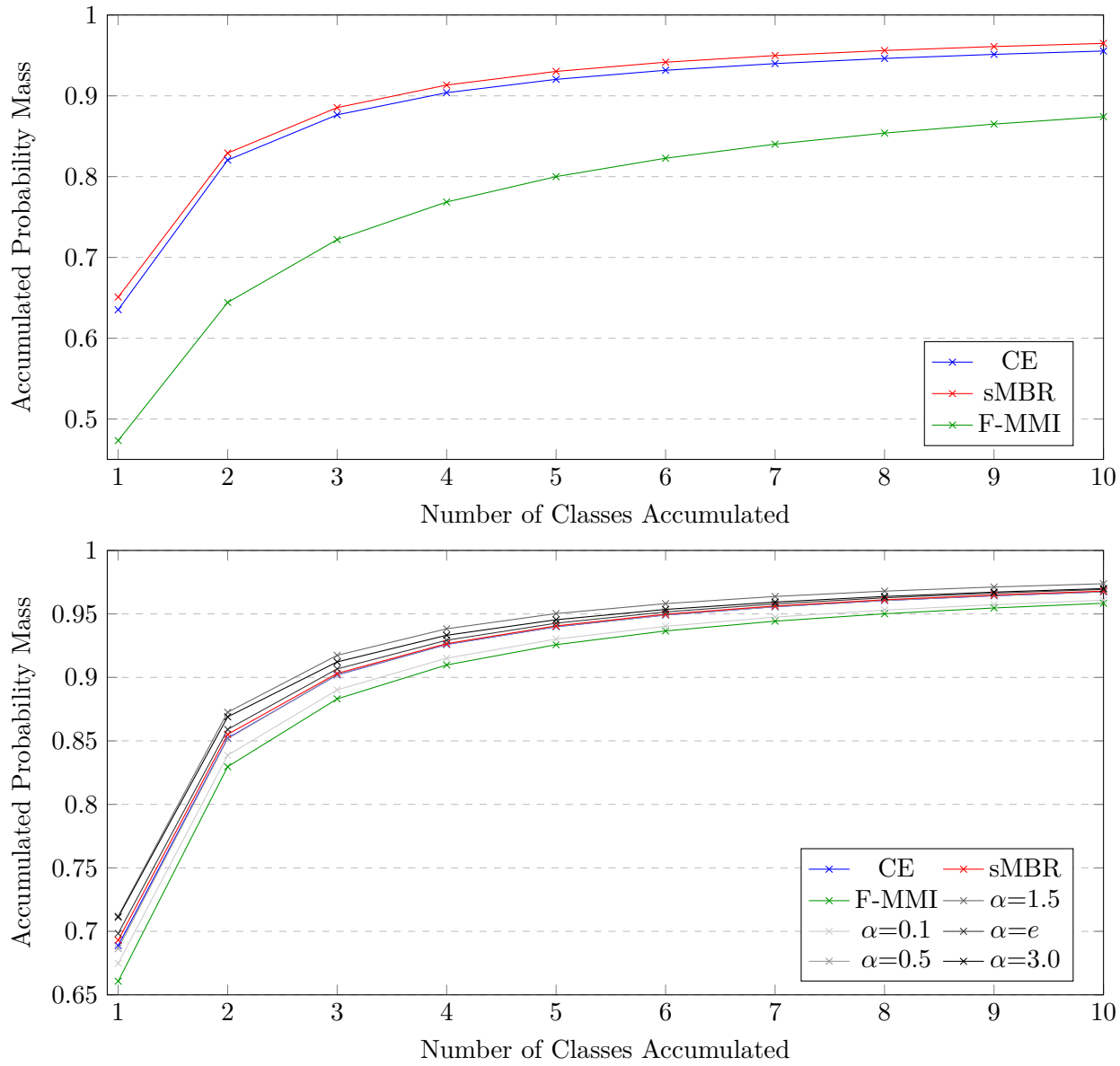

Figure 3.1: Comparison of the accumulated probability mass for the n classes with highest probability averaged over all speech frames for different training criteria (n is on the x-axis). Silence frames according to the reference alignment have been discarded for this plot. (top: Quaero, bottom: Switchboard)

Table 3.5: Comparing improvements in terms of word error rate (WER) from sequence discriminative training criteria for hybrid DNN-HMM acoustic models across different training criteria, test sets, and research groups. Column "from" contains the error rate of a model trained with cross-entropy training criterion and column "to" the error rate after training with the indicated sequence discriminative criterion. [maximum mutual information (MMI), frame-level maximum mutual information (F-MMI), lattice-free maximum mutual information (LF-MMI), boosted maximum mutual information (bMMI), and state-level Bayes risk (sMBR)]

author / publication	training criterion	test set	WER [%] from	to
this work	MMI	LBS other	10.0	9.0
	F-MMI		10.0	9.1
	sMBR		10.0	8.8
	MMI	hub5'01	14.7	13.8
	F-MMI		14.7	14.2
	sMBR		14.7	13.9
[Veselý & Ghoshal$^+$ 13]	MMI	hub5'01	19.8	18.4
	sMBR		19.8	18.0
	MPE		19.8	18.2
	bMMI		19.8	18.3
[Su & Li$^+$ 13]	MMI	hub5'00	16.2	14.1
	sMBR		16.2	14.7
	bMMI		16.2	14.0
[Xiong & Droppo$^+$ 17]	LF-MMI	hub5'00	12.9	11.9
[Saon & Sercu$^+$ 16]	sMBR	hub5'00	13.3	12.4
[Povey & Peddinti$^+$ 16]	LF-MMI	hub5'00	18.2	15.5
	sMBR		18.2	16.9
	LF-MMI	LBS other	5.0	4.3
	sMBR		5.0	4.6
[Vyas & Madikeri$^+$ 21]	LF-MMI	Babel Swahili	30.4	29.4
	LF-MMI	LBS other	8.5	7.3
[Lu & Xiao$^+$ 19]	MMI	LBS other	13.5	12.5
	sMBR		13.5	12.5
	bMMI		13.5	12.5
[Sak & Senior$^+$ 15]	sMBR	proprietary	14.0	12.9
[Saon & Sercu$^+$ 16]	sMBR	proprietary	13.9	12.7
[Wang & Su$^+$ 19]	MMI	proprietary	5.7	4.5
[Tian & Zhang$^+$ 17]	sMBR	proprietary1	2.8	2.5
		proprietary2	7.9	6.3
[Tachioka & Watanabe$^+$ 14]	bMMI	CHiME2	27.3	22.6
[Hu & Cai$^+$ 17]	LF-MMI	IAM	13.8	10.4

4. Sentence-Level Language Model Integration for Attention-Based Acoustic Models

4.1 Introduction

In this chapter we consider models where the output probability distribution is defined on a per output symbol level, independent of the time scale of the acoustic features. The alignment problem in this case is solved implicitly or explicitly by the underlying neural model. This can, for example, be done by the attention mechanism in attention-based encoder-decoder models [Bahdanau & Cho$^+$ 15].

Here, the neural model directly models the sentence-wise posterior probability $p\left(w_1^N|x_1^T\right)$ and can be used directly in the decision rule. Since only a single model is used, this approach has often been termed "end-to-end modeling" [Prabhavalkar & Hori$^+$ 24]. It has, however, been shown that, while end-to-end models are theoretically well defined, the addition of an explicit language model that has been trained on additional textual data improves the word error rate significantly [Gülçehre & Firat$^+$ 15, Toshniwal & Kannan$^+$ 18]. Most state-of-the-art attention-based ASR systems include explicit language models during recognition via log-linear model combination (*shallow fusion*) but an integration in the training phase is often not considered.

In this chapter we will focus on attention-based encoder-decoder models and will show how to consistently include external language models during training and decoding. In section 4.2 the log-linear combination method used in this chapter is presented. In section 4.3 the training criteria that arise from this integration method will be presented. Section 4.4 will detail the setup of the experiments performed in this chapter, including the model description, training, and decoding procedures. Finally, the main results of this chapter will be presented in section 4.5.

4.2 Sentence-Level Model Combination Approach

For the hybrid approach of the previous chapter, the way to integrate an (external) language model was clearly defined by the Bayes decomposition of Equation 3.2. For direct models this is much less clear, and several approaches have been explored in the past.

In this chapter, a method is used that very closely matches the way the language model was introduced for the hybrid model. Instead of the Bayes decomposition, where the language model arose from changing the conditional and the independent variable in the sentence posterior, here a log-linear model combination is used. In this way, the sentence posterior probability, the acoustic model $p\left(w_1^N\middle|x_1^T\right)$, is combined with a sentence prior model, the language model $p\left(w_1^N\right)$.

In order for the combined probability to be normalized, a normalization constant $Z(x_1^T)$ that

only depends on the input features is introduced.

$$p_{\text{comb}}\left(w_1^N|x_1^T\right) = \frac{1}{Z(x_1^T)} \cdot p\left(w_1^N\middle|x_1^T\right) \cdot p\left(w_1^N\right) \tag{4.1}$$

$$= \frac{p\left(w_1^N\middle|x_1^T\right) \cdot p\left(w_1^N\right)}{\sum_{M,v_1^M} p(v_1^M|x_1^T) \cdot p(v_1^M)} \tag{4.2}$$

This sentence-level combination was originally proposed for the shallow fusion approach [Gülçehre & Firat+ 15]. In this work the combined probability is obtained by multiplying the sentence-level probability distributions of acoustic model $p_{\text{AM}}^{\alpha}\left(w_1^N\middle|x_1^T\right)$ and language model $p_{\text{LM}}^{\beta}\left(w_1^N\right)$ and additionally applying model scales α, β.

$$p_{\text{comb}}\left(w_1^N|x_1^T\right) = \frac{p_{\text{AM}}^{\alpha}\left(w_0^N\middle|x_1^T\right) \cdot p_{\text{LM}}^{\beta}\left(w_1^N\right)}{\sum_{M,v_1^M} p_{\text{AM}}^{\alpha}(v_1^M|x_1^T) \cdot p_{\text{LM}}^{\beta}(v_0^M)} \tag{4.3}$$

$$= \frac{\prod_{n=1}^{N} p_{\text{AM}}^{\alpha}\left(w_n\middle|w_0^{n-1}, x_1^T\right) \cdot p_{\text{LM}}^{\beta}\left(w_n\middle|w_0^{n-1}\right)}{\sum_{M,v_1^M} \prod_{m=1}^{M} p_{\text{AM}}^{\alpha}\left(v_m\middle|v_0^{m-1}, x_1^T\right) \cdot p_{\text{LM}}^{\beta}\left(v_m\middle|v_0^{m-1}\right)} \tag{4.4}$$

A generalization to three or more different models is straightforward by adding more factors and exponents to the numerator and denominator respectively.

4.3 Training Criteria

4.3.1 Symbol Discriminative Cross-Entropy Training Criterion

The most widely used training criterion for attention-based acoustic models is the *cross-entropy criterion*. Here the cross-entropy of the model output with the empirical distribution over the training data $\mathcal{T}$ is used as a training criterion.

In the simplest case the acoustic model is trained independently of the language model. Then the sentence log posterior decomposes into an independent sum of symbol log posteriors.

$$F_{\text{CE}}(\theta) = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \log p(w_1^N|x_1^T) \tag{4.5}$$

$$= \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \log \prod_{n=1}^{N} p_\theta(w_n|w_0^{n-1}, x_1^T) \tag{4.6}$$

$$= \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \sum_{n=1}^{N} \log p_\theta(w_n|w_0^{n-1}, x_1^T) \tag{4.7}$$

In each position the reference symbol is discriminated against all competing symbols via the softmax in the model definition. We denote this version of cross-entropy without external language model with the abbreviation CE.

4.3.2 Sequence Discriminative Training Criteria

Maximum Mutual Information

Starting from the sentence-level cross-entropy we can find a different variant of the cross-entropy training criterion by including a language model with the sentence-level renormalization

of Equation 4.4.

$$F_{\mathrm{MMI}}(\theta) = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \log p_{\mathrm{comb}}(w_1^N | x_1^T) \tag{4.8}$$

$$= \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \log \frac{p_{\mathrm{AM}}^{\alpha}(w_1^N | x_1^T) \cdot p_{\mathrm{LM}}^{\beta}(w_1^N)}{\sum_{M, v_1^M} p_{\mathrm{AM}}^{\alpha}(v_1^M | x_1^T) \cdot p_{\mathrm{LM}}^{\beta}(v_1^M)} \tag{4.9}$$

$$\begin{aligned} = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} & \alpha \sum_{n=1}^{N} \log p_\theta(w_n | w_0^{n-1}, x_1^T) \\ & + \beta \sum_{n=1}^{N} \log p_\varphi(w_n | w_0^{n-1}) \\ & - \log \sum_{M, v_1^M} \prod_{m=1}^{M} p_\theta^{\alpha}(v_m | v_0^{m-1}, x_1^T) \cdot p_\varphi^{\beta}(v_m | v_0^{m-1}) \end{aligned} \tag{4.10}$$

Here we find three terms in Equation 4.10. The first term is the symbol-level cross-entropy (cf. Equation 4.7) for the acoustic model, the second term is the cross-entropy for the language model, and only the third term combines both models and makes the gradient of each model depend on the other.

Written in the form of Equation 4.9, this criterion resembles the maximum mutual information criterion for hybrid models as given in Equation 3.2 of the previous chapter. But here a log-linear model combination is used in place of a Bayes decomposition. We will therefore denote this training criterion the *maximum mutual information criterion* (MMI), although it is in fact the same criterion as cross-entropy with a different combination approach.

In this form, however, the criterion discriminates the reference sentence w_1^N in the numerator against all possible sentences v_1^M in the denominator. It is therefore a sequence discriminative training criterion.

Expected Error

The expected error rate (EER) based class of training criteria aims to maximize the expected accuracy of the model with respect to a given accuracy measure $A(w_1^N, v_1^M)$.

$$F_{\mathrm{EER}} = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \sum_{M, v_1^M} p(v_1^M | x_1^T)\, A(w_1^N, v_1^M) \tag{4.11}$$

The accuracy function is typically chosen to be the Levenshtein distance (L) between the two sequences. For the probability distribution, we can use the acoustic model alone or in combination with a language model. In this work a language model will always be included, therefore the full version of the criterion is given in Equation 4.12

$$F_{\mathrm{EER}}(\theta) = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \sum_{M, v_1^M} \frac{p_\theta^{\alpha}(v_1^M | x_1^T) \cdot p_\varphi^{\beta}(v_1^M) \cdot \mathrm{L}(v_1^M, w_1^N)}{\sum_{L, u_1^L} p_\theta^{\alpha}(u_1^L | x_1^T) \cdot p_\varphi^{\beta}(u_1^L)} \tag{4.12}$$

4.3.3 Denominator Approximation

Language model combination at the sentence level requires the computation of a normalization over all possible word sequences which is unfeasible. Therefore, in practice, the denominator is approximated. In this section two approximation schemes for the denominator are compared: An approximation with an n-best list, which is widely used in the literature [Prabhavalkar &

Sainath$^+$ 18, Meng & Wu$^+$ 21, Guo & Tiwari$^+$ 20], and an approximation with a trellis, which is a novel approach proposed in this work.

Approximation with n-best List

The denominator of Equation 4.9 contains a sum over all possible word sequences of all lengths, which is intractable to compute. Instead, a fixed size n is selected and only the n hypotheses with the highest score according to the decision rule in Equation 4.19 are considered in the sum. These hypotheses are selected for each training step using a GPU-based beam search implementation with fixed beam size for efficiency. If the reference word sequence is not found in the set of hypotheses, then the hypothesis with the lowest probability is replaced by the reference word sequence.

With the set of considered hypotheses as $\mathcal{B}$, Equation 4.9 becomes

$$F_{\text{MMI}} = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \log \frac{p_{\text{AM}}^{\alpha}(w_1^N | x_1^T) \cdot p_{\text{LM}}^{\beta}(w_1^N)}{\sum_{M, v_1^M \in \mathcal{B}} p_{\text{AM}}^{\alpha}(v_1^M | x_1^T) \cdot p_{\text{LM}}^{\beta}(v_1^M)}. \tag{4.13}$$

With this approximation the combined model is not properly normalized any more. Only the normalization of the underlying models and the inclusion of the reference word sequence in the denominator ensure a bound on the training criterion.

Approximation with Trellis

The approximation with an n-best list greatly reduces the number of considered hypotheses. For hybrid models, a compact representation of the hypothesis space could be achieved using lattices (see subsection 3.3.2). The acoustic and language models in this chapter, however, have an unlimited word context size, and paths in the search space cannot be recombined.

At evaluation time, approximative (or "forced") recombination has been used to obtain word lattices in the presence of language models with unlimited context size [Beck & Zhou$^+$ 19]. Here, the same approximation is done during training.

First, a history cutoff length k is selected. Then all probability distributions that share at least k common symbols of history are subjected to the approximation

$$p(w_n | w_{n-k}^{n-1}, w_0^{n-k-1}, x_1^T) \approx p(w_n | w_{n-k}^{n-1}, \hat{w}_0^{n-k-1}, x_1^T) \quad \forall w_0^{n-k-1} \tag{4.14}$$

with

$$\hat{w}_0^{n-k-1} = \arg\max_{w_0^{n-k-1}} p(w_n | w_{n-k}^{n-1}, w_0^{n-k-1}, x_1^T).$$

All probabilities in the set of probabilities with the same k common symbols of history are replaced by the highest probability from the set. This enables a recombination of paths in the search space that share at least k common symbols.

The beam search algorithm of the previous section is extended to allow for approximative recombination of paths. A fixed beam size in combination with recombinations now leads to a search space in the form of a trellis. An example of such a trellis is depicted in Figure 4.1. Special care is taken for the reference word sequence. If the reference word sequence is not found during search, the set of hypotheses with the lowest total probability is replaced by the reference word sequence. If the reference word sequence is part of the trellis and a candidate for recombination, then the reference word sequence is always selected for $\hat{w}_0^{n-k-1}$ even if other word sequences would lead to higher probabilities. This ensures that the probability of the reference word sequence, i.e. the numerator of Equation 4.9, is always present and correct.

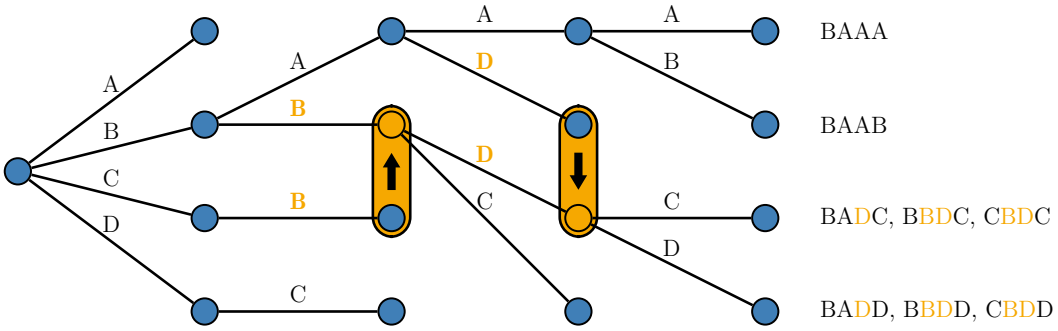

Figure 4.1: Illustration of approximative recombination with beam size $b = 4$ and history cutoff length $k = 1$. The matching histories that are eligible for recombination are highlighted in orange. The set of hypotheses covered by the search space is listed on the right. Taken from [Wynands 21].

Hypotheses that contain finished sentences are always recombined independently of the selected history cutoff length. The full implementation of this algorithm is listed in Appendix C.1.

Finally, the full sum over all word sequences is approximated by the sum over all word sequences that are contained in the trellis $\mathcal{T}$. Equation 4.9 then becomes

$$F_{\text{MMI}} = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \log \frac{p_{\text{AM}}^{\alpha}(w_1^N | x_1^T) \cdot p_{\text{LM}}^{\beta}(w_1^N)}{\sum_{M, v_1^M \in \mathcal{T}} p_{\text{AM}}^{\alpha}(v_1^M | x_1^T) \cdot p_{\text{LM}}^{\beta}(v_1^M)}. \tag{4.15}$$

4.4 Experimental Setup

4.4.1 Modeling Approach

Acoustic Models

The acoustic models used in this chapter are attention-based encoder-decoder models. They model the sentence posterior using a symbol-wise probability distribution.

$$p_{\text{AM}}\left(w_1^N \middle| x_1^T\right) = \prod_{n=1}^{N} p_\theta\left(w_n \middle| w_0^{n-1}, x_1^T\right) \tag{4.16}$$

We use a (bidirectional) long short-term memory (BLSTM) based neural acoustic model [Hochreiter & Schmidhuber 97, Rossenbach & Zeyer$^+$ 20]. We extract standard 40-dimensional MFCC features, which are perturbed with the correlated dropout mechanism from SpecAugment [Park & Chan$^+$ 19]. The encoder consists of two convolution layers with filter size $3 \times 3 \times 32$ followed by 6 BLSTM layers with 1024 units per direction. After the first two BLSTM layers, the time axis is subsampled by max-pooling with factors 3 and 2 respectively. Therefore the encoder output sequence length is reduced by a factor 6. We use dropout [Srivastava & Hinton$^+$ 14] and an auxiliary connectionist temporal classification (CTC) loss [Graves & Fernández$^+$ 06] on the encoder outputs for regularization.

The decoder uses standard attention with weight feedback [Vaswani & Shazeer$^+$ 17] and consists of a single LSTM layer followed by a linear layer with the maxout [Goodfellow & Warde-Farley$^+$ 13] activation function (both with 1000 units) and a linear layer with the softmax activation function. The output of the last layer represents the probability distribution over the vocabulary of 10k byte-pair encoding [Gage 94] tokens. The total number of parameters is 190M.[1]

[1]The model definition is available online at `https://github.com/rwth-i6/returnn-experiments/blob/master/2019-asr-e2e-trafo-vs-lstm/librispeech/base2.conv2l.specaug.curric3.config`

Language Models

In this chapter we will exclusively use neural language models in training and testing, which model the sentence probability using a symbol-wise probability distribution.

$$p_{\mathrm{LM}}\left(w_1^N\right) = \prod_{n=1}^{N} p_\varphi\left(w_n \middle| w_0^{n-1}\right) \tag{4.17}$$

For recognition, we use a transformer-based language model (*trafo*) with an embedding size of 128 followed by 24 transformer layers each with 8 attention heads. The model is trained on the transcriptions of the training corpus plus external textual data.

We also use an LSTM-based language model (*lstm*) with an embedding size of 128 followed by 4 LSTM layers with 2048 units each. The model is also trained on the transcriptions of the training corpus plus external textual data and can be used in training and recognition.

Finally, we use a language model structure that is inspired by the acoustic model decoder (*decoder-like*). We use an embedding size of 128 and a single LSTM layer with 1000 units and train it only on the transcriptions of the training data. The modeling power of this LM should therefore approximately match the language modeling power that can be learned by the acoustic decoder.

If not noted otherwise, the *decoder-like* language model is used in training and the *trafo* language model during recognition.

4.4.2 Training Procedure

With the training procedure in this chapter we aim to minimize computational effort and maximize comparability of the results. For this we have adopted a two-step training process:

- In a first step, acoustic and language models are trained independently with the cross-entropy criterion. These models serve as a common initialization for the subsequent step.
- In a second step, we continue to train the acoustic models using different training criteria and integration methods. For comparison, we also continue to train the model with the cross-entropy criterion but without any language model to get the baseline result.

In this chapter, the initial model is trained for 25 epochs and the training is continued for 5 further epochs.[2]

4.4.3 Decision Rule

Throughout the chapter the *maximum a posteriori* (MAP) decision rule

$$\hat{w}_1^{\hat{N}} = \arg\max_{N, w_1^N} \left\{ p_{\mathrm{comb}}\left(w_1^N | x_1^T\right) \right\} \tag{4.18}$$

is used. The normalization constant $Z(x_1^T)$ will not influence the result of the arg max and can be omitted. Then also one of the two model scales can be set to 1 without loss of generality, as only the ratio α/β influences the result. The full form of the decision rule used in this chapter is

$$\hat{w}_1^{\hat{N}} = \arg\max_{N, w_1^N} \left\{ \prod_{n=1}^{N} p_\theta\left(w_n \middle| w_0^{n-1}, x_1^T\right) \cdot p_\varphi^\beta\left(w_n \middle| w_0^{n-1}\right) \right\}. \tag{4.19}$$

[2] Training configs for experiments with approximative recombination are available online at `https://github.com/rwth-i6/returnn-experiments/tree/master/2022-approx-recombination`, all other experiments at `https://github.com/rwth-i6/returnn-experiments/tree/master/2020-lm-integration`

4.5 Results

4.5.1 Search Space Size and Approximation

As a first step we will analyze the effect of search space size on the performance of sequence discriminative training of attention-based acoustic models. For this we compare the n-best list approximation and the trellis approximation for different list lengths n and history cutoff lengths k.

For n-best lists, the search space size in terms of number of hypotheses is directly given by its length n. For trellises, the effective number of hypotheses contained in the trellis depends on the history cutoff length k and the number of recombinations observed during training. To estimate the effective search space size in training, we count the average number of recombinations and the average number of hypotheses on the first 1000 utterances in the LibriSpeech dev-other test set.

Acoustic Model with Unlimited Context Size

The results for the long short-term memory based acoustic model with the decoder-like language model are given in Table 4.1. The number of recombinations increases from 2 to 16 as the history cutoff length is reduced from 12 to 1. The effective number of hypotheses contained in a trellis of size 8 exponentially increases from a few tens with cutoff length 10 to thousands with cutoff length 3 to trillions at a cutoff length of 1. The probability mass that is covered by all hypotheses in the trellis is increased by three orders of magnitude from $\exp(-9.9)$ to $\exp(-7.1)$.

With a decrease in cutoff length we increase not only the search space that is covered but also the error that is incurred by the approximation in Equation 4.14. To measure the quality of the approximation, we calculate the Euclidean or $L2$ distance

$$L = \left\| p(w|w_{n-k}^{n-1}, w_0^{n-k-1}, x_1^T) - p(w|w_{n-k}^{n-1}, \hat{w}_0^{n-k-1}, x_1^T) \right\|_2 \tag{4.20}$$

between the correct probability distribution and the approximated one. The Euclidean distance is then averaged over all recombinations with the respective number of common symbols and reported in the last column in Table 4.1. A distance of 0.0 would mean no difference between the distributions and $\sqrt{2}$ would mean completely orthogonal distributions. We notice that the approximation error increases from 0.02 to 0.2 while the cutoff length is decreased.

In Table 4.2 we list the word error rates that were obtained with the long short-term memory based acoustic model that was trained with the maximum mutual information training criterion together with the decoder-like language model employing either the trellis-approximation with different history cutoff lengths k or the n-best list approximation with different list sizes n.

For the trellis approximation with a trellis size of 8 we notice that reducing the cutoff length leads to a degradation of model performance. At a cutoff length of 10 or larger, we recover the performance of infinite cutoff length, i.e. an n-best list of size 8.

For the n-best list approximation we notice no further improvements by increasing the list size over a threshold of 6. Smaller list sizes lead to some degradation, but even at a list size of 2, i.e. corrective training [Bahl & Brown$^+$ 88], we get visible improvement over the baseline cross-entropy criterion.

The benefit of the additional covered hypotheses in the trellis should be more pronounced when the search space is further confined. We therefore also test the trellis approximation with a trellis size of 4. Here we notice minimal improvements for cutoff lengths of 10 and 4, while other lengths show no change compared to an n-best list of size 4.

Table 4.1: Trellis search space statistics for an acoustic model with unlimited context length averaged over the first 1000 utterances of the LibriSpeech dev-other test set. For each history cutoff length k, the size is reported in terms of covered probability mass, number of hypotheses, number of recombinations. Additionally, the average Euclidean distance between the correct and the approximative probability distribution is reported.

history cutoff length k	log probability mass	average number of hypotheses	average number of recombinations	Euclidean distance
1	-7.1	$1.2 \cdot 10^{13}$	15.8	0.195
2	-8.0	$1.2 \cdot 10^{7}$	10.0	0.106
3	-8.5	$3.1 \cdot 10^{3}$	7.5	0.077
4	-8.8	$1.9 \cdot 10^{3}$	6.0	0.060
6	-9.2	$4.3 \cdot 10^{1}$	4.1	0.041
8	-9.3	$2.3 \cdot 10^{1}$	3.1	0.028
10	-9.5	$1.7 \cdot 10^{1}$	1.0	0.023
12	-9.6	$1.4 \cdot 10^{1}$	1.9	0.018
∞	-9.9	$0.8 \cdot 10^{1}$	0.0	–

Acoustic Model with Limited Context Size

To limit the influence of the approximation in Equation 4.14, we replace the acoustic model with unlimited context with a model that only has access to a limited number of past symbols. To this end we replace the long short-term memory layer in the decoder by a feed-forward layer that only gets the last 5 symbols as inputs. There is still an indirect dependence on previous symbols via the attention mechanism, but we argue that by limiting the decoder context, the similarity between probability distributions that share at least 5 common symbols is greatly increased. Additionally, the recurrent layer in the language model is replaced by 3 linear layers with the same context length of 5.

The search space statistics for the feed-forward-based acoustic model together with a 5-gram language model are given in Table 4.3. We see similar trends as in the case of models with unlimited context length. The number of recombinations increases from 3 to 16 as the history cutoff length is reduced from 8 to 1. The effective number of hypotheses contained in the trellis of size 8 also exponentially increases from a few tens with cutoff length 8 to thousands with cutoff length 4 to trillions at a cutoff length of 1. The probability mass that is covered by all hypotheses in the trellis is again increased by three orders of magnitude from $\exp(-9.9)$ to $\exp(-6.8)$.

Looking at the Euclidean distance, on the other hand, we see a strong difference compared to the case of acoustic models with unlimited context. For cutoff lengths below the model context length, the error of the approximation is even larger, but as we reach the context length of the model $k = 5$, the error is drastically reduced to a fraction of the previous value. At a cutoff length of 8 the distance of the limited context model is only 0.0002 compared to the distance of 0.03 for the unlimited-context model.

This is also reflected in the word error rates that can be obtained by sequence training with trellis approximation, as reported in Table 4.4. Models trained with the trellis approximation with a history cutoff length of 5 that exactly matches the model context length have better word error rates than models trained with n-best list approximation. However, the absolute value of the best word error rate obtained 6.2 is still worse than the best result from a model with unlimited context length 5.8.

Table 4.2: Comparing trellis and n-best list search space approximations with different sizes in terms of word error rate (WER) in % of a long short-term memory based acoustic model with unlimited context length trained together with the decoder-like language model and the maximum mutual information training criterion.

denominator approximation	trellis or n-best list size	cutoff length k	WER [%] dev-clean	WER [%] dev-other
no denominator	-	-	2.3	6.5
trellis	8	1	2.2	6.2
		2	2.1	6.2
		3	2.1	6.1
		4	2.1	6.0
		6	2.1	5.9
		8	2.1	5.9
		10	2.1	5.8
		12	2.1	5.8
		∞	2.1	5.8
trellis	4	1	2.2	6.2
		2	2.1	6.0
		3	2.1	6.0
		4	2.1	5.9
		6	2.1	6.0
		8	2.1	6.0
		10	2.1	5.9
		12	2.1	6.0
		∞	2.1	6.0
n-best list	2	∞	2.2	6.2
	4		2.1	6.0
	6		2.1	5.8
	8		2.1	5.8
	10		2.1	5.8
	12		2.1	5.8
	16		2.1	5.8

Conclusions: Search Space Size and Approximation

In this section we have tested different approximations of the denominator term in Equation 4.9 of the maximum mutual information training criterion. The widely used n-best list approximation was compared to a novel approach that uses approximative recombination to construct a trellis. Also, different search space sizes are compared.

The results of this section are summarized in Table 4.5. We found that the search space size does not matter much once a certain minimal threshold is passed. Models with unlimited context length are not improved by using the trellis approximation over an n-best list and are even degraded if the history cutoff length is too small. Models with limited context length show improvements when using the trellis approximation over an n-best list, if the history cutoff length matches the model context length.

We conclude that the benefit of having additional hypotheses in the trellis is outweighed by the damage of the approximative recombination necessary. In further experiments we will therefore

Table 4.3: Trellis search space statistics for an acoustic model with limited context length of 5 symbols averaged over the first 1000 utterances of the LibriSpeech dev-other test set. For each history cutoff length k, the size is reported in terms of covered probability mass, number of hypotheses, number of recombinations. Additionally the average Euclidean distance between the correct and the approximative probability distribution is reported.

history cutoff length k	log probability mass	average number of hypotheses	average number of recombinations	Euclidean distance
1	-6.8	$7.2 \cdot 10^{12}$	15.9	0.2392
2	-7.8	$3.5 \cdot 10^{7}$	9.9	0.1363
4	-8.6	$2.8 \cdot 10^{3}$	5.9	0.0842
5	-8.8	$6.3 \cdot 10^{2}$	4.8	0.0028
8	-9.3	$2.7 \cdot 10^{1}$	3.0	0.0002
∞	-9.9	$0.8 \cdot 10^{1}$	0.0	–

Table 4.4: Comparing trellis and n-best list search space approximations with different sizes in terms of word error rate (WER) in % of a long short-term memory based acoustic model with limited context length of 5 symbols trained together with a 5-gram language model and the maximum mutual information training criterion.

denominator approximation	trellis or n-best list size	cutoff length k	WER [%] dev-clean	WER [%] dev-other
no denominator	-	-	2.6	6.7
trellis	8	5	2.3	6.2
	4		2.3	6.2
n-best list	8	∞	2.3	6.4
	4		2.4	6.3

use the n-best list approximation with a size of 8.

4.5.2 Language Model Selection

It is clear that in this work we want to integrate a language model into the training of acoustic models. In this section we will investigate what kind of language model is best suited for this endeavor. There are several tuning points we will investigate in this section:

Language Model Context Length

Limiting the language model context length is an effective way to control the expressiveness and quality of the language model. On the acoustic transcriptions we trained several feed-forward-based language models with limited past context of sizes 1 (bigram), 2 (trigram), 4 (5-gram), and a long short-term memory based language model with infinite context. These models are then used in the maximum mutual information training criterion of Equation 4.9 with the training procedure described in subsection 4.4.2.

The results are shown in Table 4.6. A larger context size improves the perplexity on both the train and the development set. Also the word error rate after training with the maximum mutual

Table 4.5: Comparing the word error rate (WER) of the combined models trained with the maximum mutual information criterion and (approximative) denominator combination history limit k. WER of the model used for initialization (init) is included for reference.

model context	history limit k	dev WER[%] clean	dev WER[%] other	test WER[%] clean	test WER[%] other
limited	init	2.6	6.7	3.0	7.1
	5	2.3	6.2	2.6	6.6
	∞	2.3	6.4	2.6	6.7
unlimited	init	2.4	6.7	2.7	7.0
	12	2.1	5.8	2.3	6.5
	∞	2.1	5.8	2.3	6.5

Table 4.6: Investigating the influence of language model (LM) context length on the perplexity and word error rate (WER) achieved by training with the maximum mutual information criterion on the LibriSpeech corpus.

LM context	number of parameters	perplexity train	perplexity dev	dev WER [%] clean	dev WER [%] other
no LM	-	-	-	2.3	6.5
1	30M	192	213	2.1	6.0
2		106	144	2.1	5.9
4		93	132	2.1	5.8
∞	16M	76	112	2.1	5.8

information criterion is improved by larger context sizes.

Overfitting on the Training Data

Overfitting is usually not desired for any kind of model. However, in this case the training data for the language model is the same data that we are using when we apply the language model in the acoustic model training. Therefore, it might give additional benefits if the model has better performance. We train a long short-term memory based language model on the training transcriptions. During the training process we monitor the train and development set perplexity. We then select several checkpoints based on the amount of over- and underfitting that the model exhibits.

The results are shown in Table 4.7. We see that the model does converge after 7 epochs of training. After that, the training perplexity continues to decrease while the development set perplexity starts to grow again. This is a clear sign of overfitting. In terms of word error rate, we see that both underfitting and overfitting hurt the performance of using the model in maximum mutual information training.

Additional Training Data

If the language model is trained only on the acoustic training transcriptions, then it is better matched to the data that is used in training. Additional training data might help the model to generalize better and to improve the overall performance of the language model. We train three

Table 4.7: Investigating the influence of overfitting the training language model (LM) on the training transcripts on the perplexity and word error rate achieved by maximum mutual information training on the LibriSpeech corpus. We use an LSTM-based LM with 16M parameters.

training epochs	perplexity train	perplexity dev	dev WER [%] clean	dev WER [%] other
no LM	-	-	2.3	6.5
1	242	155	2.1	6.0
2	130	128	2.1	5.9
4	97	118	2.1	5.8
7	76	112	2.1	5.8
20	43	129	2.1	5.9
50	41	133	2.1	5.9

long short-term memory based language models. A smaller model is trained only on the training transcripts while a larger model sees additional text-only data. For comparison we also train the small model on the full set of additional training data.

The results of this section are presented in Table 4.8. We see that additional training data does improve perplexity on the development set. Even the training set perplexity is reduced by using additional data. For the word error rates, we do not see any change when using additional data or more parameters.

Table 4.8: Investigating the influence of available training data and language model (LM) size on the perplexity and word error rate (WER) achieved by training with the maximum mutual information criterion on the LibriSpeech corpus. We use an LSTM-based LM.

training data	number of parameters	perplexity train	perplexity dev	dev WER [%] clean	dev WER [%] other
no LM	-	-	-	2.3	6.5
transcripts	16M	76	112	2.1	5.8
+ ext. data	16M	94	81	2.1	5.8
	140M	52	45	2.1	5.8

Conclusions: Language Model Selection

In this section we have investigated which language model should be used for the training of acoustic models with the maximum mutual information training criterion.

According to the covering hypothesis [Schlüter & Müller$^+$ 99], a strong language model might mask errors made by the acoustic model and reduce the learning effect. In this work we have not found any evidence that reducing the language model performance by either limiting the context length or undertraining leads to any benefits.

Increasing the language model performance by adding additional text-only data and/or increasing the number of parameters does not change the word error rates either and only serves to increase the time to prepare the language model.

We found that the optimal language model to use in training is a neural model that matches

the acoustic model decoder network in both number and shape of parameters. As training data the transcriptions of the acoustic training data are sufficient.

4.5.3 Training Criteria

In this section we compare the sequence discriminative training criteria introduced in section 4.3. For this we select the optimal language model and scales based on development set word error rates individually for each training criterion.

The results are given in Table 4.9. Both sequence discriminative training criteria improve over the symbol discriminative cross-entropy baseline by about 10% relative. Between the training criteria there are only minor differences. On dev-other the maximum mutual information criterion has a slightly lower word error rate, while on test-other the expected error criterion performs best.

Table 4.9: Comparing word error rate (WER) of different training criteria on the LibriSpeech corpus. All hyperparameters were optimized individually on the respective dev sets.

training	LM	dev WER [%]		test WER [%]	
criterion	combination	clean	other	clean	other
cross-entropy	no	2.3	6.5	2.7	7.0
cross-entropy	yes	2.1	5.8	2.3	6.5
expected error	yes	2.1	5.9	2.3	6.3

Conclusions: Training Criteria

Sequence discriminative training criteria that include a language model in the training process improve the word error rate compared to the cross-entropy baseline. There is no strong evidence to prefer either the maximum mutual information or the expected error criterion over the other.

4.5.4 Comparison to Other Results in the Literature

Training Criteria

In this section we compare the results and improvements we observed when using sequence discriminative training criteria to results reported by other groups.

Table 4.10 shows an overview of relevant work in the field of sequence discriminative training for direct models. The improvements obtained in word error rate vary a lot between different test sets and groups, with the best relative improvement of 13.4% from [Chiu & Sainath$^+$ 18]. Our results fall in line with reports of other groups on the LibriSpeech test-other set between 7% and 11%.

In this work we have compared the maximum mutual information training criterion and the expected word error rate training criterion. In the literature expected word error rate seems to be the prevalent choice of training criterion. Apart from [Yang & Zhou$^+$ 23], which is from our group, only [Zhu & Huang 20] has employed the maximum mutual information training criterion.

Additionally, most works do not integrate a language model into the training process, despite the fact that [Weng & Yu$^+$ 20] and [Meng & Wu$^+$ 21] both showed that larger improvements could be achieved if a language model was included.

Language Model Selection

There has not been any investigation as to which language model should be used in the sequence training of attention-based encoder-decoder models. While investigations on Gaussian mixture

models [Schlüter & Bezrukov$^+$ 07] seem to indicate that a weaker language model should be preferred, more recent investigations on neural transducer models [Yang & Zhou$^+$ 23, Yang & Zhou$^+$ 24] agree with the work described here that limiting context length leads to degraded performance.

Search Space Size

Our findings for the influence of search space size (cf. Table 4.2) are in line with the results reported in [Prabhavalkar & Sainath$^+$ 18] that a small beam size (4 in their experiments) is sufficient to approximate the search space, and larger n-best lists do not show significant improvements. The search space approximation with a trellis is unique to this work. In [Shannon 17] a sampling-based approximation method was introduced, but [Prabhavalkar & Sainath$^+$ 18] showed that it does not yield any improvements in word error rate over the cross-entropy baseline for attention-based encoder-decoder models.

4.6 Conclusions

In this chapter we have investigated how language models can be included in the training process of attention-based encoder-decoder acoustic models via sentence-level log-linear model combination. This gave rise to a novel training criterion that we termed "Maximum Mutual Information" and was one of the first works that included an external language model in the existing expected word error rate training criterion.

To compute these training criteria a summation over all possible word sequences was needed. In this work we investigated different methods to approximate this summation.

We were the first group to show that this approach is feasible in terms of computation time and to provide a training procedure that consistently improves the word error rate of existing models.

Search Space Approximation

In subsection 4.5.1 we compare two different approximations to the sum over all sentences necessary when doing the sentence-level model combination: n-best lists and trellis. During this work we engineered an algorithm to do approximative recombination of hypotheses that share a common history prefix (Equation 4.14). We were the first group to use approximative recombination to create trellises in training.

Unlimited Context Models In Table 4.1 we showed that, depending on the selected cutoff length, the average number of hypotheses represented in the trellis can reach up to 10^{13} hypotheses, which would be intractable with n-best lists. The total probability mass captured by the trellis was increased by 3 orders of magnitude from $exp(-10)$ for an n-best list of comparable size to $exp(-7)$ for a first order trellis.

This increased search space coverage, however, did not lead to better word error rates after training. As can be seen in Table 4.2, on LibriSpeech dev-other the word error rate of the trained model is the same 5.8% for both a trellis with a cutoff length larger than 8 and the n-best list. Reducing the cutoff length only degrades the final word error rate up to 6.2% for a cutoff length of 1.

Additionally, we investigated the influence of the size of the n-best list on the model performance. In Table 4.2 we showed that increasing the number of hypotheses contained in the n-best list further than 6 does not improve the word error rate any further than 5.8%. Smaller n-best list

sizes degrade the word error rate up to 6.2% for a list that contains only the reference word sequence and the best competitor hypothesis (i.e. corrective training), which is still an improvement over the 6.5% word error rate of a model trained without any language model integration.

We therefore conclude that for attention-based encoder-decoder models, an n-best list of size 6 sufficiently captures the relevant search space and that approximative recombination into a trellis hurts model performance.

Limited Context Models For models with limited past context the recombination into a trellis can be done without an approximation. We investigated an encoder-decoder model with a context of five symbols and constructed a trellis with history cutoff length 5. Compared to an n-best list of size 8, a trellis with the same size contained on average 630 hypotheses, as can be seen in Table 4.3. The word error rate we obtained on LibriSpeech dev-other using a trellis was improved slightly from 6.4% for the n-best list to 6.2% for the trellis.

We conclude that a trellis is helpful in cases where it can be constructed without approximation. But the effect is small and enforcing the constraint of limited context length on a model degrades the word error rate more than the improvements we get from increasing the search space size with a trellis.

Language Model Selection for Training

In this work we investigated several aspects of the language model to be included in training and their influence on the perplexity of the model and the final word error rate of the acoustic model after training.

Context Size We investigated if reducing the language model context size in training influences the word error rate. In Table 4.6 we see that the word error rate on LibriSpeech dev-other when training with a neural trigram (5.9%) or bigram (6.0%) language model was increased compared to using an unlimited context language model (5.8%).

We found that limiting the context size in training hurts the performance.

Training Data Perplexity The language model we use in training is itself trained on the transcriptions of the same training data. We can therefore control the perplexity on the training data by over- or undertraining the language model. In Table 4.7 we showed that when we select the language model based on the development set perplexity, we get a train set perplexity of 76 and the best word error rate of 5.8%. If we overtrain the language model, the train set perplexity is improved to 41, but the word error rate degrades to 5.9%. If we undertrain the language model, the train set perplexity degrades to 240 and the word error rate to 6.5%.

We conclude that the language model should be selected by development set perplexity.

Additional Training Data We investigated whether adding additional external training data beyond the acoustic transcripts to the language model training data would improve the language model. In Table 4.8 we see that adding more training data and increasing the language model size improves the perplexity from 110 to 45 but has no influence on the final word error rate, which is 5.8% in both cases.

We conclude that when including a language model in the training process of an attention-based encoder-decoder acoustic model, the language model should have the same size and complexity as the acoustic model decoder and should be trained on the transcriptions of the acoustic training data.

Training Criterion

Lastly, we compared sequence-level cross-entropy and expected word error rate as two sequence discriminative training criteria with language model integration. In Table 4.9 we show that both criteria improved over the baseline without language model integration on LibriSpeech test-other from 7.0% to 6.5% for the sentence cross-entropy and to 6.3% for the expected error criterion.

While the expected error criterion performed better on the test set, the sentence cross-entropy gave the best results on the development set and both methods performed equally in the clean sub-partitions. We see no clear winner between these criteria.

Verdict

Including a language model in the training process for attention-based encoder-decoder acoustic models is a worthwhile approach to improve the acoustic model performance and reduce word error rate without influencing the run-time cost of the model in any way. The optimal language model for this has the same size and complexity as the acoustic model decoder and should be trained on the transcriptions of the acoustic training data. The exact training criterion does not matter much as long as a language model is included.

4.7 Joint Work and Prior Publications

The results of this chapter were already published in [Michel & Schlüter$^+$ 20a] and [Wynands & Michel$^+$ 22]. The initial models with unlimited context length were designed and pretrained by Mohammad Zeineldeen, based on prior work of Albert Zeyer. The models with limited context length were designed by Mohammad Zeineldeen and Alexander Glushko.

The shallow fusion approach for attention-based encoder-decoder models was introduced in [Gülçehre & Firat$^+$ 15]. Using it as a training criterion was proposed by the author. Approximative recombination to create lattices out of search trees was proposed in [Beck & Zhou$^+$ 19]. Using it during training to approximate the sum in the denominator was proposed by the author.

The maximum mutual information and expected error criteria were implemented by the author. The initial implementation of the recombination in training was done by the author and then later refined by Nils-Philipp Wynands. The experiments with trellises were done by Nils-Philipp Wynands under the supervision of the author. All other experiments were done by the author.

Table 4.10: Comparing improvements in terms of word error rate (WER) from sequence discriminative training criteria with or without external language model (LM) for end-to-end acoustic models across different training criteria, test sets, and research groups. Column "from" contains the error rate of a model trained with cross-entropy training criterion and column "to" the error rate after training with the indicated sequence discriminative criterion [maximum mutual information (MMI) or minimum expected word error rate (mWER)]. Some works use a lattice-free version of the maximum mutual information criterion (LF-MMI) as an auxiliary loss for the encoder, in which case the language model combination does not directly influence the decoder parameters.

test set	author / publication	training criterion	incl. LM	WER [%] from	WER [%] to
LBS other	this work	MMI	yes	7.0	6.5
		mWER		7.0	6.3
	[Tian & Yu$^+$ 22]	LF-MMI	encoder	5.4	5.0
	[Yang & Zhou$^+$ 23]	MMI	yes	4.6	4.1
		mWER		4.6	4.1
LBS other (mismatched)	[Meng & Wu$^+$ 21]	mWER	no	14.5	14.2
			yes	14.5	11.7
LBS clean-mix	[Kanda & Meng$^+$ 21]	mWER	no	15.6	14.2
self recorded	[Zhu & Huang 20]	MMI	no	39.1	38.4
Aishell-1	[Tian & Yu$^+$ 22]	LF-MMI	encoder	5.1	5.0
	[Tian & Yu$^+$ 23]	LF-MMI	encoder	4.7	4.5
Aishell-2	[Tian & Yu$^+$ 23]	LF-MMI	encoder	5.9	5.8
Hub5'00	[Weng & Cui$^+$ 18]	mWER	no	19.0	17.8
				12.6	11.9
proprietary (Amazon)	[Guo & Tiwari$^+$ 20]	mWER	no	10.0	9.6
proprietary	[Prabhavalkar & Sainath$^+$ 18]	mWER	no	6.6	6.2
(Google)	[Chiu & Sainath$^+$ 18]	mWER	no	6.7	5.8
proprietary (Tencent)	[Weng & Yu$^+$ 20]	mWER	no	5.7	5.2
			yes	5.7	5.0
			no	15.3	15.2
			yes	15.3	14.9
proprietary (Microsoft)	[Meng & Wu$^+$ 21]	mWER	no	12.0	11.4
			yes	12.0	11.1
proprietary (Microsoft)	[Lu & Meng$^+$ 21]	mWER	no	16.3	15.9

5. Symbol-Level Language Model Integration for Attention-Based Acoustic Models

5.1 Introduction

In the previous chapter we have shown how a language model can be integrated into the training of attention-based encoder-decoder acoustic models with a sentence-level combination. All approaches had in common that for the normalization of the model combination an intractable sum over all possible word sequences had to be computed. Some kind of approximation was always necessary for this.

In this chapter we will take a similar approach and again consider attention-based encoder-decoder models but want to avoid the costly summation. By breaking down the combination from sentence-level to the level of one individual symbol or modeling unit, we will transform one sum over all sentences into many sums over the vocabulary. This will reduce the computational complexity and make computing the correct normalization term feasible.

In section 5.2 we will present the modified model combination approach at symbol-level and section 5.3 will contain the resulting training criteria. Section 5.4 will show the setup of the experiments performed in this chapter, including the description of the models, as well as the training and decoding procedures. Finally, in section 5.5 we will present the results we obtained in this work.

5.2 Symbol-Level Model Combination Approach

The symbol-level model combination was introduced in [Michel & Schlüter$^+$ 20a] and termed *local fusion*. Here, the combined probability is first decomposed into symbol-level probabilities and then, for each position, a log-linear combination with separate normalization is used. The normalization constants $Z_n(w_0^{n-1}, x_1^T)$ now depend not only on the acoustic data x_1^T but also on the symbol history w_0^{n-1}.

$$p_{\text{comb}}\left(w_1^N|x_1^T\right) = \prod_{n=1}^{N} p_{\text{comb}}\left(w_n|w_0^{n-1}, x_1^T\right) \tag{5.1}$$

$$= \prod_{n=1}^{N} \frac{1}{Z_n\left(w_0^{n-1}, x_1^T\right)} \cdot p_{\text{AM}}^{\alpha}\left(w_n\middle|w_0^{n-1}, x_1^T\right) \cdot p_{\text{LM}}^{\beta}\left(w_n\middle|w_0^{n-1}\right) \tag{5.2}$$

$$= \prod_{n=1}^{N} \frac{p_\theta^{\alpha}\left(w_n\middle|w_0^{n-1}, x_1^T\right) \cdot p_\varphi^{\beta}\left(w_n\middle|w_0^{n-1}\right)}{\sum_v p_\theta^{\alpha}\left(v\middle|w_0^{n-1}, x_1^T\right) \cdot p_\varphi^{\beta}\left(v\middle|w_0^{n-1}\right)} \tag{5.3}$$

A generalization to combine three or more different models can be achieved straightforwardly by adding more factors and exponents to the numerator and denominator, respectively.

Symbol-Dependent Model Scales

The acoustic and language model scales α and β weight the importance of the respective models in the combination. This is currently done via a single weight per model. Depending on the currently hypothesized word, a different weighting might be needed, as e.g. some words are acoustically distinct while other words are clearly distinguished by the context. For this, symbol-dependent model scales $\alpha(w)$ and $\beta(w)$ are introduced. The combined model now reads

$$p_{\text{comb}}\left(w_1^N|x_1^T\right) = \prod_{n=1}^{N} \frac{p_\theta^{\alpha(w_n)}\left(w_n\middle|w_0^{n-1}, x_1^T\right) \cdot p_\varphi^{\beta(w_n)}\left(w_n\middle|w_0^{n-1}\right)}{\sum_v p_\theta^{\alpha(v)}\left(v\middle|w_0^{n-1}, x_1^T\right) \cdot p_\varphi^{\beta(v)}\left(v\middle|w_0^{n-1}\right)} \tag{5.4}$$

5.3 Training Criteria

Cross-Entropy

As a baseline to compare the model combination experiments against, the cross-entropy training criterion as defined in Equation 4.7 of chapter 4 is repeated here:

$$F_{\text{CE}}(\theta) = \sum_{(w_1^N, x_1^T)\in\mathcal{T}} \sum_{n=1}^{N} \log p_\theta(w_n|w_0^{n-1}, x_1^T) \tag{5.5}$$

Symbol-Level Language Model Combination

If a language model should be included in the training process without the additional computational requirements and approximations needed to compute the denominator that was required in the last chapter, the symbol-level combination method of Equation 5.4 can be used. The starting point is again the sentence-level cross-entropy, but now it is first decomposed into symbol-wise contributions before the log-linear combination is applied.

$$F_{\text{LF}}(\theta) = \sum_{(w_1^N, x_1^T)\in\mathcal{T}} \log p_{\text{comb}}(w_1^N|x_1^T) \tag{5.6}$$

$$= \sum_{(w_1^N, x_1^T)\in\mathcal{T}} \log \prod_{n=1}^{N} p_{\text{comb}}(w_n|w_0^{n-1}, x_1^T) \tag{5.7}$$

$$= \sum_{(w_1^N, x_1^T)\in\mathcal{T}} \sum_{n=1}^{N} \log \frac{p_\theta^\alpha\left(w_n\middle|w_0^{n-1}, x_1^T\right) \cdot p_\varphi^\beta\left(w_n\middle|w_0^{n-1}\right)}{\sum_v p_\theta^\alpha\left(v\middle|w_0^{n-1}, x_1^T\right) \cdot p_\varphi^\beta\left(v\middle|w_0^{n-1}\right)}. \tag{5.8}$$

The normalization now consists of multiple sums over the vocabulary instead of a single sum over all sentences. The combinatorial complexity to enumerate all summands is reduced from $\mathcal{O}(V^N)$ to $\mathcal{O}(V \cdot N)$, with V the size of the vocabulary and N the length of the sentence. This is the same complexity as the normalization of each individual model in the softmax output and should not significantly influence the training speed.

Here the reference word at each position is discriminated against all other words, both explicitly through the renormalization in the denominator and implicitly through the normalization of the individual models. Symbol-level language model combination therefore yields a symbol-discriminative training criterion.

5.4 Experimental Setup

5.4.1 Modeling Approach

We use the exact same attention-based encoder-decoder acoustic model and the LSTM/transformer-based language models as described in the previous chapter subsection 4.4.1. If not noted otherwise, the *lstm* language model is used in training and the transformer *trafo* language model in decoding.

5.4.2 Training Procedure

We use the same two-step training procedure as described in subsection 4.4.2 of the previous chapter, where the acoustic model is first pretrained with the cross-entropy criterion and then the training is continued with the criterion that is to be investigated. For the experiments of Sections 5.5.1 and 5.5.2, the model was pretrained for 4 epochs and the training was continued for 8 additional epochs. [1] In all other experiments the model was pretrained for 12 epochs and continued for 5 additional epochs.[2]

5.4.3 Decision Rules

In this chapter again the *maximum a posteriori* (MAP) decision rule

$$\hat{w}_1^{\hat{N}} = \arg\max_{N, w_1^N} \left\{ p_{\text{comb}}\left(w_1^N | x_1^T\right) \right\} \tag{5.9}$$

is used. If now symbol-level log-linear model combination (Equation 5.3) is used, then the normalization constant $Z_n(w_0^{n-1}, x_1^T)$ depends on the argument of the arg max, i.e. the symbol history, and must be kept. The sum in the denominator also implies that all model scales must be kept. Here we arrive at the decision rule

$$\hat{w}_1^{\hat{N}} = \arg\max_{N, w_1^N} \left\{ \prod_{n=1}^{N} \frac{p_\theta^\alpha\left(w_n \middle| w_0^{n-1}, x_1^T\right) \cdot p_\varphi^\beta\left(w_n \middle| w_0^{n-1}\right)}{\sum_v p_\theta^\alpha\left(v \middle| w_0^{n-1}, x_1^T\right) \cdot p_\varphi^\beta\left(v \middle| w_0^{n-1}\right)} \right\}. \tag{5.10}$$

Constant Denominator Approximation

To avoid the additional effort of calculating the normalization constant for each position and to tune one additional scale, the constant denominator approximation can be applied. Here it is assumed that the normalization does not differ much for different word histories.

$$Z_n(w_0^{n-1}, x_1^T) \approx Z_n(x_1^T) \qquad \forall n, w_0^{n-1} \tag{5.11}$$

Then the normalization constants can again be left out of the decision rule and one scale can be dropped without loss of generality. The resulting decision rule is equivalent to the decision rule for sentence-level model combination, Equation 4.19

$$\hat{w}_1^{\hat{N}} = \arg\max_{N, w_1^N} \left\{ \prod_{n=1}^{N} p_\theta\left(w_n \middle| w_0^{n-1}, x_1^T\right) \cdot p_\varphi^\beta\left(w_n \middle| w_0^{n-1}\right) \right\}. \tag{5.12}$$

[1] The configs for automatic scale learning are available online at `https://github.com/rwth-i6/returnn-experiments/tree/master/2022-scale-learning`

[2] All other training configs are available online at `https://github.com/rwth-i6/returnn-experiments/tree/master/2020-lm-integration`

5.5 Results

5.5.1 Automatic Learning of Model Scales

Until now the acoustic and language model scales α and β have been treated as hyperparameters that were given in the experimental conditions. They are typically tuned using a grid search and by doing multiple recognition steps with fixed scales. Those scales are then selected, which give rise to the lowest word error rate on some development set. We will refer to this method as *manual* tuning.

In this section we want to instead treat the scales as part of the model parameters and try to learn them using our training criteria and a gradient-based update rule. This has the advantage that only a single run is needed to find the best scales up to an arbitrary precision. A disadvantage is that training in general is slower than a single recognition step.

If not noted otherwise, the training procedure is as follows. First, acoustic and language model are trained separately using the standard cross-entropy criterion (Equation 5.5). Then they are combined using randomly initialized (mean 1.0 and variance 0.5) scaling exponents. While the exponents are trained, the other model parameters are kept constant.

For the training criterion we need to make sure that the resulting combined model is properly normalized. If we wanted to maximize the score of a log-linear combination directly without renormalization, as in the common shallow fusion approach, then the optimal solution would be to move one or all scales towards infinity. We therefore use symbol-level language model combination (Equation 5.8) to enforce proper normalization. We also investigate using the expected error criterion (Equation 4.12), which matches more closely the manual optimization process of minimizing the word error rate.

In Table 5.1 we report the results of automatic scale learning. For the decision rule we apply the constant denominator approximation and therefore only the ratio of β/α matters for the word error rate. We see that the cross-entropy criterion fails to find the optimal scales. Optimization on the training data leads to a too small language model scale, while optimization on both dev sets yields too large scales.

The expected error training criterion outperforms the cross-entropy criterion and finds the same optimum as the manual grid search process. The automatically learned scales even outperform the manually tuned ones by a small margin as they are optimized with higher numerical precision.

Table 5.1: Word error rates (WER) and values of automatically learned acoustic (α) and language model (β) scales trained on the LibriSpeech corpus using cross-entropy or expected error training criterion. Scales with manual tuning are given as a reference.

scale training					dev WER [%]		test WER [%]	
criterion	set	α	β	$\frac{\beta}{\alpha}$	clean	other	clean	other
manual	dev-other	2.77	1.00	0.36	2.9	8.3	3.2	9.0
cross entropy	train	1.24	0.26	0.21	3.0	8.7	3.2	9.3
	dev-clean	1.10	0.51	0.46	3.1	8.4	3.5	9.3
	dev-other	1.00	0.50	0.50	3.1	8.5	3.6	9.7
expected error	train	3.41	1.19	0.35	2.8	8.2	3.2	8.9
	dev-clean	3.14	0.89	0.28	2.9	8.3	3.2	9.1
	dev-other	2.82	1.46	0.52	3.1	8.3	3.5	9.2

Conclusions: Automatic Learning of Model Scales

With the expected error training criterion we were able to recover the same scales that were found by manually tuning. This is not surprising as this is a convex optimization problem with a single parameter. The computational effort for manual tuning was much smaller compared to the automatic procedure.

We conclude that for single scalar scales, the manual tuning procedure is sufficient, especially if one has a good idea or prior knowledge about the location of the optimum. The automatic tuning procedure might still be useful if the number of scaling parameters is increased. By our estimation, a naive grid search of 4 or more scaling parameters will be more costly than the automatic procedure.

5.5.2 Symbol-Dependent Model Scales

For symbol-dependent model scales as introduced in subsubsection 5.2 the number of parameters to optimize is as large as the vocabulary size, which is 10k in our experiments. It is infeasible to optimize this many parameters "by hand", therefore the automatic scale learning approach of the previous section is used.

We use the same training criteria, training procedure, and decision rule as in the previous section, with the only difference being the number of parameters to be optimized. The results are reported in Table 5.2.

The cross-entropy criterion is again worse than the expected error criterion. Optimization on the development sets fails for the cross-entropy criterion with up to doubled error rates compared to the symbol-independent baseline. Optimization on the training data yields results that slightly improve over the symbol-independent baseline.

The expected error criterion outperforms the cross-entropy criterion. Optimizing the symbol-dependent model scales on the train set leads to 7% relative improvement over the symbol-independent scales. Optimization on the development sets gives a larger improvement of 11% on the respective development set but a slight degradation on all other sets. The average values of acoustic and language model scales do not differ when optimized on different datasets with the expected error criterion.

Table 5.2: Word error rates (WER) and average values of automatically learned symbol-dependent acoustic $\alpha(w)$ and language model $\beta(w)$ scales trained on the LibriSpeech corpus using cross-entropy or expected error training criterion. Scalar scales with manual tuning are given as a reference.

scale training					dev WER [%]		test WER [%]	
criterion	set	$\bar{\alpha}$	$\bar{\beta}$	$\frac{\bar{\beta}}{\bar{\alpha}}$	clean	other	clean	other
manual	dev-other	2.77	1.00	0.36	2.9	8.3	3.2	9.0
cross-entropy	train	1.49	0.63	0.45	2.8	8.0	3.1	8.5
	dev-clean	1.50	1.13	0.84	6.0	17.7	12.2	20.7
	dev-other	1.40	1.36	0.90	8.5	8.1	9.1	15.1
expected error	train	1.50	0.63	0.45	2.7	7.8	3.0	8.4
	dev-clean	1.50	0.64	0.45	2.5	7.9	3.1	8.5
	dev-other	1.50	0.64	0.45	2.7	7.4	3.1	8.5

Joint Training of Scales and Models

So far we have only optimized the scaling exponents of the individual models while the other model parameters were kept constant. The automatic scale learning approach does allow us to train both the model parameters and the scales at the same time.

For this experiment we train the models individually with the cross-entropy criterion as before. The scales are initialized with the values found in the previous section. We then continue the training and allow the scales, the model parameters, or both to be trained.

The results are presented in Table 5.3. Training the acoustic model weights in the presence of a language model leads to 10% relative improvement over the simple log-linear combination without training the acoustic model weights. The same word error rates are achieved for both symbol-dependent and symbol-independent model scales. Also, jointly training the model parameters with the model scales does not change the final word error rate.

Table 5.3: Word error rates (WER) after joint training of symbol-independent and symbol-dependent scales and the acoustic model on the LibriSpeech corpus with the cross-entropy criterion.

scale version	train		dev WER [%]		test WER [%]	
	AM	scales	clean	other	clean	other
symbol-independent	no	no	2.9	8.3	3.2	9.0
	no	yes	3.0	8.7	3.2	9.3
	yes	no	2.6	7.5	2.8	8.0
	yes	yes	2.6	7.6	2.8	8.2
symbol-dependent	no	yes	2.8	8.0	3.1	8.5
	yes	no	2.6	7.5	2.8	8.0
	yes	yes	2.6	7.5	2.8	7.9

Conclusions: Symbol-Dependent Model Scales

Symbol-dependent model scales that are optimized automatically with the expected error criterion show improvements over using a single symbol-independent scaling parameter. Training the acoustic model parameters jointly with a language model in the process, further improves the word error rate. However, when training the acoustic model parameters, the benefit of symbol-dependent model scales vanishes completely.

We conclude that the modeling power of symbol-dependent scales can also be learned by the acoustic model. In the following we will continue to use symbol-independent model scales which are manually tuned and then kept constant.

5.5.3 Decision Rule

In subsection 5.4.3 two decision rules were presented. One decision rule respects the normalization term that is included by symbol-level model combination, while the other drops it in favor of a sentence-level normalization. For the latter it is easier to optimize the scaling parameters as only the relative weight matters. The former better matches the training criterion used during the training of the acoustic models. In this section we want to investigate the impact of the decision rule on the performance of models also trained with symbol-level language model combination.

The results are presented in Table 5.4. The decision rule with symbol-level normalization has an optimum at acoustic model scale 0.6 and language model scale of around 0.38, which means a

relative scale between 0.62 and 0.66. The decision rule with sentence-level normalization has the optimum at a relative scale of 0.54.

Table 5.4: Comparing word error rates (WER) in % for decision rules with symbol-level and sentence-level normalization for an acoustic model trained with symbol-level language model combination on the LibriSpeech corpus. Error rates are reported on the dev-other development set.

decision rule		relative scale LM/AM									
normalization	AM scale	0.3	0.4	0.5	0.54	0.58	0.62	0.66	0.7	0.8	0.9
symbol	0.4	7.9	7.3	7.2	7.2	7.3	7.4	7.5	7.7	8.4	9.1
	0.5	7.8	7.0	6.7	6.7	6.6	6.6	6.6	6.7	7.1	7.8
	0.6	7.6	7.0	6.6	6.5	6.5	**6.4**	**6.4**	6.5	7.0	7.4
	0.7	7.8	7.2	6.8	6.7	6.7	6.7	6.8	6.7	7.1	7.5
	0.8	8.0	7.5	7.2	7.1	7.1	7.1	7.1	7.2	7.5	7.9
	1.0	8.6	8.1	7.9	7.9	7.9	7.9	8.0	8.1	8.3	8.8
sentence	1.0	7.4	6.7	6.2	**6.1**	6.2	6.3	6.3	6.3	6.7	7.4

Conclusions: Decision Rule

The optimal relative scale with sentence-level normalization is significantly lower than the optimal relative scale with symbol-level normalization. This implies that the relative importance of the models does depend on the decision rule. The word error rate obtained with the sentence-level normalized decision rule is lower than the word error rate obtained by the symbol-level normalized decision rule. This, together with the reduced tuning effort necessary to find this optimum, clearly speaks to preferring the sentence-level decision rule. We will exclusively use the decision rule with sentence-level normalization in the following.

5.5.4 Language Model Exchange

One of the advantages of the log-linear model combination approach with symbol-level combination is, that the individual models can be exchanged after training. This is in contrast to the deep fusion technique, where the language model hidden states are fed into the acoustic model and the acoustic model is adapted to a specific set of hidden states.

But with log-linear combination the model could be adapted to the one specific language model it has seen in training and perform worse if paired with other models after training. In this section we want to investigate if such an exchange of language models after training is possible without degraded performance.

For this we train three acoustic models: One without language model, one paired with a long short-term memory (LSTM) based language model, and a third one paired with a transformer-based language model. After the training converges, we switch out the language models for recognition and compare the resulting word error rates.

The results are presented in Table 5.5. For the cross-entropy baseline both language models give good improvements over not using a language model for recognition. The transformer outperforms the LSTM by 8% relative and leads to 31% improvement over not using a language model.

When a language model is used in training, the word error rate for recognition without language model degrades, but recognition with language model is improved. For the LSTM-based language model, the transformer still outperforms the LSTM by 8% relative and now leads to 55% improvement over not using a language model. Using the transformer-based language model in training

leads to slightly worse word error rates compared to using the LSTM-based language model, but the relative difference between using the LSTM language model and using the transformer language model in recognition remains at 8%.

Table 5.5: Comparing the effect of using a different language model for training and testing on the LibriSpeech corpus. Models are trained with the cross-entropy criterion.

		dev WER [%]	
training LM	testing LM	clean	other
-	-	3.9	10.6
	LSTM	2.8	7.9
	TRAFO	2.6	7.3
LSTM	-	5.3	13.7
	LSTM	2.5	6.6
	TRAFO	2.2	6.1
TRAFO	-	4.0	11.0
	LSTM	2.5	6.8
	TRAFO	2.4	6.2

Conclusions: Language Model Exchange

We conclude that exchanging the language model after the training phase is indeed possible, and the improvements of a better language model are independent of the language model used in training. We did not notice any benefit of using matched language models in training and testing. We therefore conclude that the best available language model should always be used during recognition regardless of the model used in training.

5.5.5 Comparison to Sequence Discriminative Training

Training Speed

The main intention of including a language model with symbol-level log-linear combination was to avoid the overhead of the denominator computations necessary for the sequence discriminative training criteria of chapter 4. We give up the sequence discriminative nature of the training criterion and only retain the language model integration. We hope to retain some of the improvements in word error rate while losing the increased training time.

In Table 5.6 we compare the time it takes to complete a single sub-epoch of 50h of training data. We see that the symbol-level language model combination is only minimally slower than using the cross-entropy criterion without language model, where both have a real-time factor of 1/60. The training criteria with sentence-level language model combination are slower by a factor of 5 and operate at a real-time factor of around 1/12.

Word Error Rate

The fastest training is of no use if the word error rates are not improved. In Table 5.7 we compare the word error rates obtained after training with sequence discriminative training criteria with the word error rates obtained by training with symbol-level language model combination. We notice that despite the symbol-level integration not being sequence discriminative, it does achieve the same word error rate improvements of 15% relative on the development sets as the

Table 5.6: Training speed comparison for sequence-level and symbol-level language model (LM) combination. Training time in minutes per sub epoch of 50h of data from the LibriSpeech corpus.

training criterion	LM combination	time [min]	realtime factor
cross-entropy	no LM	48	0.016
	symbol-level	51	0.017
	sentence-level	249	0.08
expected error	sentence-level	252	0.08

sentence-level combination. On the test sets the improvement is only slightly behind the sentence-level combination. The expected error criterion is slightly worse compared to cross-entropy with language model.

Table 5.7: Comparing word error rates (WER) achieved by sequence-level and symbol-level language model (LM) combination on the LibriSpeech corpus. All hyperparameters were optimized individually on the respective dev sets.

training criterion	LM combination	dev WER [%]		test WER [%]	
		clean	other	clean	other
cross-entropy	no LM	2.6	7.3	2.8	7.8
	symbol-level	2.2	6.1	2.5	6.7
	sentence-level	2.2	6.1	2.4	6.4
expected error	sentence-level	2.3	6.3	2.5	6.8

Conclusions: Comparison to Sequence Discriminative Training

We found that integrating a language model into the training process does help improve the word error rates. We did not find any significant differences in the final word error rate comparing the symbol-level integration of this chapter to the sentence-level integration of chapter 4.

Comparing the impact on the training speed, we found that sentence-level integration leads to a 5× increase in training time needed, while the symbol-level integration does not significantly change the training speed compared to the cross-entropy training without language model.

We therefore conclude that it is always advisable to include a language model in training with symbol-level normalization for the cross-entropy training criterion.

5.5.6 Comparison to Other Results in the Literature

Word Error Rates

In this section we compare the improvements we observed when integrating a language model into the training process by log-linear combination with symbol-level normalization to results for other symbol-level integration techniques reported by other groups. Symbol-level integration techniques encompass all integration methods that change some aspect in the training phase of the acoustic model and that do not include the sum over all possible sentences that is characteristic of the sequence discriminative training criteria of the previous chapter.

The results are summarized in Table 5.8. There have been many studies proposing different integration methods with mixed results. [Toshniwal & Kannan$^+$ 18] failed to see any improvement

when using cold or deep fusion, while other works report relative improvements in the single-digit range [Kim & Shangguan$^+$ 21, Cabrera & Liu$^+$ 21]. Other proposed approaches also exhibit significantly lower relative improvements except in cases of weak baselines [Bai & Yi$^+$ 19] or test sets that were designed to have domain mismatch [Shan & Weng$^+$ 19].

Other methods to incorporate the language model knowledge have been proposed, such as employing the language model as a teacher in a student-teacher approach [Hinton & Vinyals$^+$ 15], or including the language modeling data in the training to train the acoustic model in a multi-task fashion. However, these approaches also showed only minor improvements.

Language Model Exchange

We have shown that with the symbol-level combination approach it was possible to exchange the language model after training. Some of the other proposed combination methods theoretically also allow the language model to be exchanged after the training phase.

In [Kim & Shangguan$^+$ 21] and [Sriram & Jun$^+$ 18] it was confirmed that such an exchange was indeed possible for the cold fusion approach. They also find that the best available language model for the target text should always be used for recognition, but in their case, a domain shift was involved that might bias towards this finding.

Decision Rule Approximation

There are no other results in the literature since this approximation is tied to the symbol-level language model combination approach that is proposed in this work.

Symbol-Dependent Model Scales and Scale Learning

We have shown that it was possible to automatically learn the model combination scales of normal, symbol-independent and also symbol-dependent model scales. Automatically learned symbol-dependent model scales were also employed in [Hoffmeister & Liang$^+$ 09] for log-linear combination of multiple acoustic and a language model during lattice rescoring. In [Chang & Lahiri$^+$ 15] multiple language models were combined with scales that were not only symbol-dependent, but also dependent on the full history of symbols. [Variani & Riley$^+$ 22] proposed to make the interpolation weights dependent on the acoustic data x_1^T. The word error rate improvements are shown in Table 5.9. This work reports the biggest relative improvement by making the scales symbol-dependent.

We have also shown that the improvements go away if the acoustic model is trained jointly with the scales. No other work has investigated this effect.

5.6 Conclusions

In this chapter we have investigated a method to speed up the language model integration for attention-based encoder-decoder models by going from a sentence-level integration to a symbol-level integration approach. Instead of renormalizing the combined sentence posterior of acoustic and language model over all possible sentences, we first decomposed the probability into a sequence of token-wise conditioned probabilities and then were able to execute the exact normalization over the symbol vocabulary.

Constant Denominator Approximation

If the symbol-level combination approach is carried over to recognition, then the decision rule needs to also contain symbol-level renormalization of acoustic and language model (Equa-

tion 5.10). This leads to additional cost, which is undesirable. In this work we investigated the *constant denominator approximation* (Equation 5.12), which drops the additional renormalization during recognition and leads to a decision rule that resembles the sentence-level combination approach.

Comparing the mathematically exact decision rule that matches the symbol-level language model combination in training with the decision rule that follows from sentence-level combination, we found that the sentence-level decision rule requires less tuning effort, since only one relevant scale has to be tuned. Additionally, we found in Table 5.4 that the best word error rate without approximation of 6.4% was 5% relative worse compared to the 6.1% we obtained using the approximation.

Therefore, we conclude that the constant denominator approximation should always be used in recognition.

Language Model Exchange

One common shortcoming of many other language model combination techniques that rely on combining hidden representations is that the language model cannot be exchanged later should a better language model become available. Our proposed approach does not suffer explicitly from this, but we investigate whether there is any performance penalty for exchanging the language model after training.

In Table 5.5 we see that the word error rate mostly depends on the language model used in recognition. Using a strong language model improves the word error rate from 6.6% to 6.1% compared to a weaker language model. The choice of language model in training has less influence on the final word error rate, and using a weaker language model seems to be slightly advantageous, at a word error rate of 6.1% compared to 6.2% for having the same strong language model during training and recognition.

We therefore conclude that the best matching language model for the test set should always be used in recognition, while a smaller language model trained on the training transcriptions is sufficient for use in training.

Symbol-Dependent Scaling Exponents

One extension of this work towards deeper language model integration techniques is to make the scaling exponents of acoustic and language model dependent on the current symbol. This increases the number of parameters from 2 to twice the size of the vocabulary (e.g. 20k). It is infeasible to tune this number of parameters manually, therefore we introduced a gradient descent-based automatic learning procedure based on an objective function.

Automatic Learning of Symbol-Independent Scaling Exponents In Table 5.1 we see that by using a back-propagation-based approach with the expected error training criterion, we can recover the same scales and word error rates as were found by manual tuning. Using the cross-entropy training criterion instead yields different optimal scales, which lead to a degraded word error rate of 9.3% compared to the manual result of 9.0%. This is to be expected as both the manual tuning and the expected-error training criterion directly aim to optimize the word error rate whereas the cross-entropy criterion does not.

Automatic Learning of Symbol-Dependent Scaling Exponents In Table 5.2 we see that symbol-dependent scales optimized with the expected error training criterion improve the word error rate to 8.4% compared to 9.0% for symbol-independent scales. The best word error rates on the test sets were achieved by using all of the training data for scale optimization. Using dev sets instead

improved the word error rate on the respective dev-set (e.g. from 7.8% to 7.4% on dev-other) but degraded all other error rates by 0.1%.

Joint Learning of Scaling Exponents and Acoustic Model Since both, the training of the acoustic model parameters and the learning of the scaling exponents use the same gradient descent-based update method, we can try to jointly train both sets of parameters. In Table 5.3 we see that jointly training the acoustic model with a language model and symbol-dependent scaling exponents improves the word error rate further to 8.0%. However, using symbol-independent scaling exponents yields the exact same word error rate.

We therefore conclude that symbol-dependent scaling exponents are feasible to learn automatically and do improve the word error rate of the final model combination. However, the same improvement can also be learned by the acoustic model parameters if they are jointly trained with the scales and a language model. There is therefore no reason to use symbol-dependent scaling exponents, and we recommend jointly training with symbol-independent scaling exponents for simplicity.

Comparison to Sequence Discriminative Training

The novel approach presented in this chapter directly competes with the sentence-level combination technique presented in chapter 4. We compare the time needed for training and the word error rate the final model achieves. The recognition time in both approaches is the same thanks to the constant denominator approximation.

Training Time As can be seen in Table 5.6, the time spent training for one sub-epoch of 50h of training data was reduced from 250 minutes for the sentence-level combination approach to 50 minutes for the symbol-level combination approach. With this speedup of a factor of 5, the symbol-level combination approach is almost on par with the baseline training approach without language model combination.

Word Error Rate In terms of word error rate, the symbol-level combination approach is on par with the sentence-level combination approach, as can be seen in Table 5.7. On the LibriSpeech dev sets, the word error rate of symbol- and sentence-level combination is exactly equal, while on the test sets, the sentence-level combination has an advantage of 6.4% versus 6.7% for the symbol-level combination. Both approaches significantly improve over the model trained without a language model, which achieved 7.8% word error rate.

Verdict

We proposed a novel approach to combine the acoustic model with a language model at training time. This symbol-level combination captures most, if not all, of the improvements to the word error rate that can be achieved by sentence-level combination, while maintaining the original training complexity and speed of training the acoustic model without a language model.

This approach is the most flexible and does not constrain the acoustic model to a particular language model during recognition. With the constant denominator approximation, there is also no computational overhead during recognition. We see no reason not to adapt to this training approach immediately.

5.7 Joint Work and Prior Publications

The results of this chapter were already published in [Meyer & Michel$^+$ 22] for the automatic learning of scaling exponents and the symbol-dependent model scales and in [Michel & Schlüter$^+$ 20a] for all other results. The initial models were designed and pretrained by Nick Rossenbach, based on prior work of Albert Zeyer.

The log-linear model combination approach with symbol-level normalization was designed, implemented, and tested by the author. The automatic scale learning and symbol-dependent model scales were designed by the author and implemented and tested by Felix Meyer under the supervision of the author. All other experiments were done by the author.

Table 5.8: Comparing improvements in terms of word error rate (WER) from symbol-level language model (LM) combination during training of end-to-end acoustic models across different combination approaches, test sets, and research groups. Column "from" contains the error rate of independently trained models and column "to" the error rate with the indicated integration approach.

test set	author / publication	integration method	LM change possible	WER [%] from	WER [%] to
LBS other	this work	log-linear comb.	yes	7.8	6.7
	[Kim & Shangguan$^+$ 21]	cold fusion	yes	8.6	8.3
				7.0	6.8
	[Cho & Watanabe$^+$ 19]	cold fusion	yes	16.6	17.1
		cell control	no	16.6	16.2
	[Wang & Sainath$^+$ 21]	multi-task train	no	10.3	10.3
LBS clean	[Wang & Sainath$^+$ 21]	multi-task train	no	9.7	9.3
hub5'00	[Toshniwal & Kannan$^+$ 18]	deep fusion	no	21.1	21.7
		cold fusion		21.1	21.8
Aishell-1	[Shan & Weng$^+$ 19]	cold fusion	yes	9.8	8.8
		component f.		9.8	8.7
	[Bai & Yi$^+$ 19]	student-teacher	no	9.9	8.3
	[Bai & Yi$^+$ 21]	student-teacher	no	5.9	5.8
Aishell-2	[Bai & Yi$^+$ 21]	student-teacher	no	6.6	6.5
Babel Swahili	[Cho & Watanabe$^+$ 19]	cold fusion	yes	56.2	52.9
		cell control	no	56.2	50.7
proprietary (Tencent)	[Shan & Weng$^+$ 19]	cold fusion	yes	25.0	20.4
		component f.		25.0	17.5
proprietary (Google)	[Cabrera & Liu$^+$ 21]	cold fusion	yes	24.2	22.4
		early shallow f.		24.2	24.4
		early cold f.		24.2	22.7
proprietary (Google)	[Cabrera & Liu$^+$ 21]	cold fusion	yes	14.4	14.0
		early shallow f.		14.4	14.5
		early cold f.		14.4	14.2
proprietary (Google)	[Toshniwal & Kannan$^+$ 18]	deep fusion	no	5.3	5.5
		cold fusion		5.3	5.3
				BLEU [%]	
WMT ET-EN	[Stahlberg & Cross$^+$ 18]	cold fusion	yes	17.9	17.8
		simple f. prenorm		17.9	19.0
		simple f. postnorm		17.9	18.6

Table 5.9: Comparing improvements in terms of word error rate (WER) from applying symbol-dependent model combination weights across different test sets and research groups. Column "from" contains the error rate of scalar model weights and column "to" the error rate with the indicated number of symbol-dependent combination weights.

test set	author / publication	number of scaling factors	WER [%] from	to
LBS other	this work	10k	9.0	8.4
	[Variani & Riley$^+$ 22]	unknown	10.5	9.3
GALE mandarin	[Hoffmeister & Liang$^+$ 09]	7k	14.5 12.7	14.2 12.5
proprietary (Google)	[Chang & Lahiri$^+$ 15]	unknown	12.1	11.6

6. Internal Language Model Correction for Attention-Based Acoustic Models

6.1 Introduction

Attention-based encoder-decoder acoustic models directly model the output word sequence. For this, they take into account all previously emitted symbols at each position. Therefore they have the capability to learn about the structure of the language, similar to what is encoded in a language model. This is the reason why they can be successfully used without an explicit language model. The *internal language model* that is learned in the decoder network takes over this task.

If they are combined with an external language model during recognition, a model combination between the internal language model of the acoustic model and the external stand-alone language model happens. Now, since the external language model is trained on more data, it should have a stronger weight in the combination. But the internal language model is tightly integrated into the acoustic decoder and no separate weight to the language modeling part of the decoder can be given. Therefore the optimal weight of the model combination is determined based on the acoustic/language model trade-off (which would favor a larger weight on the acoustic model) and the internal/external language model trade-off (which would favor a larger weight on the external language model). The resulting weights would be a compromise and sub-optimal for both trade-offs.

The solution proposed in the literature is to estimate the internal language model component of the acoustic model and to down-weight it against the acoustic model component during recognition. Several internal language model estimation methods have been proposed and tested successfully. They will be discussed in section 6.2.

The conjecture of this chapter is that including a language model in training will make the model focus more on the acoustic modeling component than the internal language modeling component. The improvements found by including a language model in training would then be related to the improvements obtained by internal language model correction.

In this chapter we will implement and compare some of the internal language model estimation and correction methods. We will investigate the influence of including a language model in training on the internal language model of the acoustic decoder. And finally, we will show how both methods interact if combined together.

6.2 Internal Language Model Estimation

True Internal Language Model

It is possible to write down exactly the correct internal language model that is included in any acoustic model by integrating out over all possible inputs.

$$p_{\text{trueLM}}\left(w\middle|w_0^{n-1}\right) = \sum_{\{x_1^T\}} p_{\text{AM}}\left(w\middle|w_0^{n-1}, x_1^T\right) \cdot p\left(x_1^T\right)$$

This is of course intractable to compute in practice and an approximation method is needed.

Proxy Language Model

The proxy language model approach, often called the "density ratio approach" in the literature [McDermott & Sak+ 19], tries to train a separate set of parameters that should mimic the behavior of the internal language model of the decoder.

$$p_{\text{proxyLM}}\left(w\middle|w_0^{n-1}\right) = p_\varphi\left(w\middle|w_0^{n-1}\right)$$

The neural architecture is matched with the decoder architecture in terms of number of parameters and topology, and the training data would consist exclusively of the transcriptions of the acoustic model training data. The "decoder-like" language model described in subsection 3.3.1 is such a proxy language model.

This proxy language model is independent of any specific trained acoustic model. This is an advantage as it can be reused for all acoustic models in an ensemble of experiments, but it is also a disadvantage as by design it cannot catch the specific features of one specific acoustic model.

Constant Encoder Language Model

The constant encoder language model approaches make use of the specific architecture of the acoustic model decoder. A typical acoustic model decoder uses an attention module that computes a weighted average over a range of encoder states. This makes the decoder depend on the acoustic input.

If this module is replaced by a constant C, then the acoustic model decoder becomes independent of the acoustic input and now represents a language model

$$p_{\text{constLM}}\left(w\middle|w_0^{n-1}\right) = p_\theta\left(w\middle|w_0^{n-1}, C\right).$$

There are several approaches to choosing the constant:

- Zero input: The acoustic input is replaced by a constant zero vector. The encoder and attention module are computed normally [Das & Sunkara+ 23].
- Zero encoder state: The encoder or attention output is replaced by a constant zero vector [Meng & Parthasarathy+ 21].
- Average encoder state: The encoder output is averaged over a portion of the training data. This average is chosen as a constant during recognition [Zeineldeen & Glushko+ 21].

This approach has the advantage that it uses the actual acoustic model parameters to estimate its internal language model. The disadvantage is that the acoustic model decoder was never trained in this way. It is therefore not guaranteed that the resulting model is a working language model in practice.

Encoder Prediction Language Model

The encoder prediction language model approach [Zeineldeen & Glushko$^+$ 21] aims to combine the advantages of the constant encoder and the proxy language model. Instead of a constant, the attention module is replaced by a different module that, based on the previous words, tries to predict a matching attention output that can be used by the acoustic model decoder to predict the next word.

$$p_{\text{predLM}}\Big(w\Big|w_0^{n-1}\Big) = p_\theta\Big(w\Big|w_0^{n-1}, C_\varphi(w_0^{n-1})\Big)$$

The encoder prediction module C consists of a small recurrent network with a separate set of trainable parameters φ. After acoustic model training is finished, the acoustic encoder and the attention module are replaced by the encoder prediction module. The ensemble is then trained on the acoustic transcriptions with the language model cross-entropy objective function. Here only the prediction module parameters φ are updated and the acoustic model parameters θ are kept constant.

6.3 Training Criteria

In this chapter the same versions of the cross-entropy training criterion as presented in the previous chapters are used:

- without language model: cf. Equation 4.7
- symbol-level combination: cf. Equation 5.8
- sentence-level combination: cf. Equation 4.9

In this chapter, in addition to the acoustic model, the parameter φ of the internal language model correction model or the proxy language model might also be trained. In this case the cross-entropy criterion is used with the acoustic training data transcriptions only:

$$F_{\text{CE}}(\varphi) = \sum_{(w_1^N, x_1^T)\in\mathcal{T}} \sum_{n=1}^{N} \log p_\varphi(w_n|w_0^{n-1}) \tag{6.1}$$

$$F_{\text{CE}}(\varphi) = \sum_{(w_1^N, x_1^T)\in\mathcal{T}} \sum_{n=1}^{N} \log p_\theta\Big(w_n\Big|w_0^{n-1}, C_\varphi(w_0^{n-1})\Big) \tag{6.2}$$

6.4 Experimental Setup

6.4.1 Modeling Approach

Acoustic Model

The acoustic models used in this chapter are attention-based encoder-decoder models. We extract 80-dimensional log-mel features from the audio and apply SpecAugment dropout [Park & Chan$^+$ 19]. The encoder comprises a front-end consisting of a convolution layer with 32 filters of size 3×3 with relu activation function and factor 2 max-pooling. This is followed by two convolutions with filter size $64 \times 3 \times 3$ and relu activation function. These use a stride in the time dimension of 3 and 2 respectively so that the acoustic signal is down-sampled by a factor of 6 in total. The encoder then follows with 12 conformer blocks as described in [Zeyer & Bahar$^+$ 19].

The decoder uses mlp-attention with weight feedback and consists of a single long short-term memory layer followed by a linear layer with maxout activation function (both with 1024 units)

and a linear layer with softmax activation function. The output of the last layer represents the probability distribution over the vocabulary of 10k byte-pair encoding tokens. The total number of parameters is 100M.[1]

Language Models

The same external language models as in Chapters 4 and 5 are used. They are described in subsubsection 4.4.1. If not noted otherwise, the *decoder-like* language model is used in training and the *trafo* language model is used during recognition.

For the encoder prediction internal language model correction method a prediction module is used. This consists of a single long short-term memory layer with 50 units and a linear layer to project the output to the appropriate dimension. The total number of parameters is 160k.

6.4.2 Training Procedure

The training procedure closely follows the procedures described in Sections 4.4.2 and 5.4.2. The initial model is trained for 100 epochs. During the first 2 epochs, the model is pretrained by increasing the number of conformer blocks from 2 to 12 and increasing the size of the conformer blocks from half to full size.

The training of the initial model is then continued for 10 further epochs with one of the training criteria listed in section 6.3. After the training is finished, four checkpoints with the best score on the cross validation set are selected and averaged. The averaged model is used for recognition.

If an internal language model correction model is used, it is trained individually for each acoustic model used for recognition. The model is trained for 2 epochs on the acoustic transcriptions.

6.4.3 Decision Rule

In this chapter the maximum a-posteriori decision rule is used exclusively with sentence-level log-linear combination. If a model is trained with symbol-level normalization, we apply the constant denominator approximation. The decision rule is the same as presented in subsection 5.4.3 (Equation 5.12), except that one further model is added to correct the internal language model (ILM). The full decision rule is therefore:

$$\hat{w}_1^{\hat{N}} = \arg\max_{N, w_1^N} \left\{ \prod_{n=1}^{N} p_\theta\left(w_n \middle| w_0^{n-1}, x_1^T\right) \cdot p_\varphi^\beta\left(w_n \middle| w_0^{n-1}\right) \cdot p_{\mathrm{ILM}}^{-\gamma}\left(w_n \middle| w_0^{n-1}\right) \right\} \tag{6.3}$$

where β is the external language model scale and γ the internal language model correction scale.

6.5 Results

6.5.1 Comparison of Internal Language Model Estimation Methods

In a first step we want to compare different internal language model estimation methods. For this we train a baseline model with the cross-entropy objective function until convergence. Then different estimation methods are used to generate an internal language model correction model.

In Table 6.1 we report the perplexity on the training and development sets along with the word error rates obtained by doing recognition with an external language model and this internal language model prediction approach.

[1] The complete set of recipes for this chapter is available online at `https://github.com/rwth-i6/i6_experiments/blob/main/users/michel/experiments/conformer_att_2022/librispeech_960/configs/seq_train.py`

Table 6.1: Comparing perplexity of different internal language model estimation methods and the word error rates (WER) achieved by using this method for internal language model correction on the LibriSpeech corpus. The baseline acoustic model is used.

estimation method	perplexity train	perplexity dev	WER [%] dev clean	WER [%] dev other
none	-	-	1.9	4.3
proxy	76	112	1.8	3.9
zero	230	243	1.8	4.0
average	210	216	1.8	3.9
prediction	102	110	1.8	3.8

The perplexity of the constant encoder correction models is more than doubled compared to the perplexity of the correction models that were actually trained as a language model. All correction methods improve the word error rate, but the difference between the methods is much smaller than would be expected from the perplexity numbers.

The best method in terms of both perplexity and word error rate is the encoder state prediction method. The worst method is the constant zero encoder method. Between those two lie the constant average encoder and the proxy model method. While the proxy model method has a lower perplexity on par with the encoder prediction method, the average encoder method wins in terms of word error rate.

Conclusions: Comparison of Internal Language Model Estimation Methods

Perplexity is a measure of how well a given language model can predict a given text. When the task is to accurately represent the internal language modeling component that is learned by the acoustic model, then perplexity can be misleading, and the word error rate should be used to select the best correction method.

The encoder prediction model method is the best of the tested methods and, if not noted otherwise, will be used in the following.

6.5.2 Internal Language Model Suppression through Training

In the previous chapters we have seen that integrating a language model in the training process via log-linear combination improves the word error rate of the trained system. Additionally we saw (cf. Table 5.5) that without any external language model, those acoustic models showed degraded performance.

We now want to investigate whether including a language model in training influences the internal language model of the acoustic model decoder. To this end, we take the baseline acoustic model from before (CE) and train two additional models while including a language model via log-linear combination with both symbol-level normalization (LF) and sentence-level normalization (MMI). We then compare the perplexities of the estimated internal language models.

In Table 6.2 we see that if a language model is included in training, all internal language model estimation methods lead to increased perplexities.

If the language model is included with symbol-level normalization, the encoder state prediction models perplexity is increased by 25%, the average encoder state perplexity is increased by 70%, and the zero-encoder perplexity is more than doubled.

If the language model is included with sentence-level normalization, the encoder state prediction perplexity is increased, but only by 20%, which is less than in the symbol-level case. Both

Table 6.2: Comparing the perplexity of different internal language model estimation models across different acoustic models trained with and without a language model (LM) on the LibriSpeech corpus.

LM integration	perplexity train			perplexity dev		
	zero	avg.	pred.	zero	avg.	pred.
no LM	230	210	102	243	216	110
symbol	502	356	128	519	368	136
sentence	575	737	116	568	693	132

constant encoder perplexities are further increased compared to no or symbol-level language model integration. Compared to the baseline without language model, the zero encoder perplexity is increased by 134% and the average encoder perplexity is increased by a factor of 2.2.

Conclusions: Internal Language Model Suppression through Training

We found that including a language model in the training of acoustic models has a strong influence on the internal language model that is implicitly learned by the decoder. This is true for both symbol-level and sentence-level integration.

Since all perplexities are increased significantly if the acoustic model is trained together with a language model, we conclude that the addition of a language model in training suppresses the internal language model of the acoustic model.

6.5.3 Combining Internal Language Model Suppression and Correction

In the previous sections we have found that correcting for the internal language model of the acoustic decoder helps to improve the word error rate of the speech recognition system. Also, including a language model in the training process helps by suppressing the internal language model component.

We now want to investigate whether both effects can be combined. To this end, we take the acoustic models from the previous section and evaluate the word error rate with and without internal language model correction. The results are presented in Table 6.3.

Table 6.3: Comparing word error rates (WER) and model combination scales of different acoustic models trained with and without a language model (LM) and the effect of internal language model correction.

training LM integration	clean sets				other sets			
	LM scales		WER [%]		LM scales		WER [%]	
	ext.	int.	dev	test	ext.	int.	dev	test
no LM	0.34	-	1.9	2.1	0.34	-	4.3	4.6
	0.40	0.30	1.8	1.9	0.60	0.38	3.8	4.4
symbol-level	0.42	-	1.8	1.9	0.48	-	3.9	4.3
	0.42	0.18	1.8	1.9	0.54	0.12	3.8	4.2
sentence-level	0.44	-	1.7	2.0	0.44	-	3.9	4.5
	0.44	0.16	1.7	1.9	0.54	0.24	3.7	4.4

Scales

Without internal language model correction, the optimal external language model scale β is higher if the model was trained together with a language model. While the cross-entropy model has an optimal scale of 0.34, the optimal scales for the models trained with language model integration are in the range of 0.42 to 0.48, which is 24% to 41% higher.

When internal language model correction is used, the optimal internal language model correction scale γ is lower if the model was trained together with a language model. While the optimal internal language model correction scale for the baseline model is at 0.3 (clean) and 0.38 (other) respectively, for the models trained together with a language model it is in the range of 0.12 to 0.24, which is a reduction by 32% to 68% relative. The scales for the external language model, on the other hand, are hardly affected.

Word Error Rates

Training together with a language model improves the word error rate without internal language model correction. But if internal language model correction is applied, the word error rates of models trained with and without a language model approach each other.

For sentence-level language model combination, out of the improvement that the internal language model correction yields for the baseline model, about half is obtained by including a language model in training, and using internal language model correction on top only yields the other half of the improvement.

For the symbol-level language model combination, almost all of the improvement is obtained by including the language model, and employing the internal language model correction does not improve much on top of this. The final word error rate is then very close to the baseline with internal language model correction. Only on the test-other test set does the symbol-level method improve more than the others.

Conclusions: Combining Internal Language Model Suppression and Correction

Improvements from training an acoustic model together with a language model and thereby suppressing the internal language model of the decoder are strongly correlated with the improvements obtained by internal language model correction. In our experiments, the final word error rate after internal language model correction was very close, independent of whether a language model was included in training or not. Further, the improvements gained by only suppressing the internal language model in training were lower than what could be obtained by applying a correction. We conclude that including a language model in training via log-linear combination is unnecessary if internal language model correction can be performed during recognition.

6.5.4 Comparison to Other Results in the Literature

Comparing Internal Language Model Estimation Methods

In this section we compare the improvements we observed by language model correction to the results reported by other groups. Table 6.4 shows an overview of the methods employed and results obtained. The individual relative improvements vary a lot between methods, models, and test conditions, between 0% and up to 32% relative improvement was achieved.

Two of the most prevalent methods are the proxy language model method and the zero encoder method. When we compared these methods, the zero encoder method was preferable. The same conclusion was found in most of the other work.

Some of the other groups [Liu & Ma+ 22, Peng & Liu+ 22] tried to estimate the correct encoder constant or even an encoder prediction model from the data. The results agree with our findings that this approach is preferable to the data-agnostic zero encoder approach.

Internal Language Model Correction and Sequence Training

Only a few works combine internal language model correction with sequence discriminative training where a language model is included in the training process. [Meng & Wu+ 21] unfortunately does not report improvements from internal language model correction for the cross-entropy baseline.

The results of [Yang & Zhou+ 24] support our findings that the improvements from internal language model correction capture most of the improvements of sequence discriminative training. In [Lu & Meng+ 21], however, it was found that the same relative improvements could be achieved with and without sequence discriminative training, showing that the effects are complementary.

Internal Language Model Correction Scales

We found that the optimal scale for the external language model was influenced by internal language model correction. In [Liu & Ma+ 22] and [Zheng & An+ 22] the sum of the (positive) external model scale and (negative) internal language model correction scale was almost always the same as the original language model scale without correction, i.e. the external language model scale was increased by the same amount that was used to correct for the internal language model. This is a much stronger connection than was observed in our work, where the increase was smaller, or even nonexistent in the case of models that were already trained with a language model. The reported scales in [Zeineldeen & Glushko+ 21] support our findings.

Internal Language Model Perplexities

We found that including a language model in the acoustic model training increased the internal language model perplexity by a significant amount depending on which estimation method was chosen. In [Yang & Zhou+ 24] it was found that sequence discriminative training reduces the perplexity of the internal language model if blank labels are included in the computation. The authors attribute this to a reduction in blank label probability through sequence training.

6.6 Conclusions

In this chapter we have investigated the interdependence between integrating a language model in the training process via log-linear model combination and the internal language model representation that is implicitly learned by the attention-based encoder-decoder acoustic model.

Internal Language Model Estimation Methods

In Table 6.1 we compared four methods to estimate the internal language model representation: Training a separate language model as a proxy for the internal language model – also known as the density ratio method, zeroing out the encoder contributions to the acoustic model, providing the average encoder contribution to the acoustic model, and training an encoder state estimation model.

In terms of perplexity, the separately trained proxy language model yielded the best language modeling performance, with a perplexity of 79 on the train set and 112 on the dev sets. This was closely followed by the encoder state estimation model with a perplexity of 102 on the train set and 110 on the dev sets. The other approaches had perplexities well over 200.

In terms of word error rate, the encoder state estimation model performed best, reducing the word error rate from 4.3% to 3.8%, whereas the proxy language model only reduced to 3.9%.

We therefore recommend using the available data and the acoustic model decoder parameters to train an encoder state estimation model that completes these parameters to produce a self-contained language model as a means of estimating the implicit language model that is learned by the acoustic model decoder parameters.

Internal Language Model Suppression

We compared the perplexities of internal language model correction models estimated for acoustic decoders that have been trained together with a language model, either via sequence discriminative sentence-level combination or with the symbol-level combination approach in Table 6.2. We observed significantly higher perplexity values than those estimated from a model trained without an additional language model. For the zeroing-out method the perplexities were doubled from 250 to 500 and for the encoder state estimation method the perplexities were increased by 14% to 25% depending on the integration method.

We conclude that one part of the gains we observed by including a language model in the acoustic training process stems from suppressing the internal language model of the decoder.

Combining Internal Language Model Suppression and Correction

Since both language model integration in training and internal language model correction seem to rely on correcting the language modeling bias of the decoder parameters, we investigated whether they can be combined.

In Table 6.3 we found that if the acoustic model was combined with a language model in training, the relative importance of the internal language model correction model was significantly reduced. The optimal scaling exponent for the correction model on the LibriSpeech dev-clean set was reduced from 0.3 for acoustic models trained without language model to 0.16 for models with sentence-level integration and to 0.18 for models with symbol-level integration.

Comparing the word error rates after training with a language model and internal language model correction was inconclusive. On test-clean all approaches land on the same word error rate of 1.9%, whereas on test-other only sentence-level combination and training without a language model share the same word error rate of 4.4%, and the symbol-level combination approach seems to improve further to 4.2%.

We took this as a sign that most of the improvement in word error rate from including a language model in training comes from suppressing the internal language model and that correcting for the internal language model can fully recover this effect.

Verdict

In this chapter we found that the improvements from internal language model correction and from training the acoustic model together with a language model stem from very similar sources and hardly complement each other.

Since sentence-level language model integration in training is very complex and costly in terms of computation, we recommend skipping it in favor of internal language model correction during recognition. Symbol-level language model integration did not incur much overhead in training, so our recommendation is to use this paired with internal language model correction whenever possible.

6.7 Joint Work and Prior Publications

The improved baseline model and recipe were created by Mohammad Zeineldeen. The language model used for recognition was trained by Kazuki Irie, while the language model used in training and as a proxy language model was trained by the author. The acoustic model trained with sentence-level language model integration was trained by Minh-Nghia Phan under the supervision of Zijian Yang. The acoustic model trained with symbol-level language model integration was trained by the author. All recognition experiments, internal language model estimations, and perplexity calculations were done by the author.

The internal language model estimation methods were already published in [Zeineldeen & Glushko$^+$ 21, McDermott & Sak$^+$ 19, Meng & Parthasarathy$^+$ 21]. The baseline model was already published in [Zeineldeen & Zeyer$^+$ 24], but none of the other results of this chapter were published prior to this work.

Table 6.4: Comparing improvements in terms of word error rate (WER) from applying internal language model (LM) correction for end-to-end acoustic models across different internal language model estimation approaches, test sets, and research groups. Column "from" contains the error rate without correction and column "to" the error rate after correcting with the indicated estimation approach.

test set	author / publication	internal LM method	WER [%] from	WER [%] to
LBS other	this work	enc. estimation	4.6	4.4
		+ seq. train	4.5	4.4
	[Zhou & Zheng$^+$ 22]	enc. estimation	5.5	4.6
		zero enc.	5.5	4.8
		average enc.	5.5	4.9
		proxy lm	5.5	5.2
	[Meng & Kanda$^+$ 21]	proxy lm	5.3	5.1
		zero enc.	5.3	4.9
		+ multi task train	5.3	3.7
	[Liu & Ma$^+$ 22]	enc. estimation	4.7	4.4
		constant enc.	4.7	4.4
	[Yang & Zhou$^+$ 24]	enc. estimation	4.9	4.3
		proxy lm	4.9	4.7
		zero enc.	4.9	4.3
		+ seq. train	4.3	4.3
	[Variani & Riley$^+$ 22]	HAT (zero)	10.5	9.9
LBS other (mismatched)	[Meng & Parthasarathy$^+$ 21]	proxy lm	13.4	12.9
		zero enc.	13.4	12.3
	[Meng & Wu$^+$ 21]	zero + seq. train	11.7	11.4
	[Liu & Ma$^+$ 22]	enc. estimation	7.4	6.0
		zero enc.	7.4	7.1
		constant enc.	7.4	6.3
TEDLium2 (mismatched)	[Zhou & Zheng$^+$ 22]	enc. estimation	16.4	15.0
		zero enc.	16.4	14.4
		average enc.	16.4	14.6
		proxy lm	16.4	14.0
SEAME [Lyu & Tan$^+$ 10]	[Peng & Liu$^+$ 22]	enc. estimation	15.0	14.5
		const enc.	15.0	14.8
ASRU'19 [Shi & Feng$^+$ 20]	[Peng & Liu$^+$ 22]	enc. estimation	11.4	10.9
		const enc.	11.4	11.5
proprietary (MSFT)	[Meng & Kanda$^+$ 21]	proxy lm	13.0	12.9
		zero enc.	13.0	12.4
		+ multi task train	13.0	11.6
proprietary (MSFT)	[Meng & Wu$^+$ 21]	zero + seq. train	11.1	10.4
proprietary (MSFT)	[Lu & Meng$^+$ 21]	HAT (zero)	15.1	14.3
		+ seq. train	14.9	14.1

7. Scientific Achievements

In this thesis we have researched different ways of improving acoustic models by a tighter integration of language models into the training process. In this chapter we present the main contributions and findings of this thesis.

Sequence Discriminative Training of Hybrid Acoustic Models

For hybrid models, in chapter 3 we proposed a novel training criterion called frame-level maximum mutual information. We found that this yields comparable word error rates to the existing criteria state-level Bayes risk and sentence-level maximum mutual information (Table 3.4). The acoustic models trained with this criterion exhibited a less sharp probability distribution compared to models trained with the state-level Bayes risk criterion (Figure 3.1).

Sentence-Level Language Model Integration for Attention-Based Acoustic Models

For attention-based encoder-decoder acoustic models we proposed in chapter 4 to integrate a language model into the training process using log-linear combination of the sentence-level probabilities. We were the first group to show that this approach is feasible and we provided a training procedure to consistently improve the word error rate of existing models.

Furthermore, in this work we analyzed the search space requirements for calculating the sentence-level normalization term of sequence discriminative training criteria for encoder-decoder models. We have shown that a small n-best list is sufficient and that even corrective training with just one competitor hypothesis improves the model (Table 4.2). During this work we engineered an algorithm (Appendix C) to do approximative recombination of hypotheses that share a common history prefix and showed that this can increase the search space coverage by several orders of magnitude (Table 4.1).

We also confirmed that the best choice of language model for this integration is one that matches the acoustic model decoder in number of parameters and topology (Table 4.6) and was trained on the acoustic training data transcriptions (Table 4.8). We have thereby shown that for encoder-decoder models, limiting the language model capacity does not lead to larger improvements from sequence discriminative training as more acoustic model errors are being corrected (Table 4.7).

Symbol-Level Language Model Integration for Attention-Based Acoustic Models

We were the first group to use symbol-level language model combination for attention-based encoder-decoder models, as outlined in chapter 5. We have shown that this approach greatly reduces the training time (Table 5.6) and complexity (Equation 5.8) of training the acoustic model together with a language model while obtaining the same relative word error rate improvements for the final models (Table 5.7).

In recognition we were able to simplify this approach by showing that the symbol-level normalization can be omitted during recognition (Table 5.4), effectively reducing the combination to

sentence-level (Equation 5.12). This not only led to improved recognition speeds, but also reduced the tuning effort and improved the word error rate of the final system (Table 5.4). Additionally we showed that acoustic models trained using this approach can be combined with a different language model in recognition without incurring a loss in word error rate from the mismatched language model (Table 5.5).

We introduced symbol-dependent model scales that were estimated automatically using a gradient descent method with an optimization function (Equation 5.4). We showed that symbol-dependent scales improve the word error rate compared to symbol-independent scales (Table 5.2), but the same improvement could be realized by jointly training the acoustic model with the language model using symbol-independent scales (Table 5.3).

Language Model Correction for Attention-Based Acoustic Models

In chapter 6 we confirmed that correcting for the internal language model that is implicitly learned by the acoustic model decoder is helpful in recognition. We compared different approaches (section 6.2) and found that using the decoder parameters together with an encoder-state estimation model consistently yields the best word error rates (Table 6.1).

By comparing internal language model perplexities we have shown that when an external language model is included in the training of acoustic models, the internal language model is suppressed by the external language model (Table 6.2).

We showed that combining both approaches, language model integration and internal language model correction, does not combine the improvements of those approaches (Table 6.3). We concluded that the internal language model correction already mostly covers the same improvements that would otherwise be achieved by suppressing the internal language model.

A. Training Corpora

A.1 LibriSpeech

The LibriSpeech corpus was released in 2015 by the Center for Language and Speech Processing & Human Language Technology of the Johns Hopkins University [Panayotov & Chen$^+$ 15]. It is part of a collection of audio books that were collected in the LibriVox project [McGuire 05]. These are public domain books that are read by volunteers and are available under the Creative Commons Attribution 4.0 license [CC4 13].

The audio was aligned to the text data and filtered for mismatch or unclear pronunciations. The resulting corpus consists of 960 hours of transcribed audio for training plus 4 held-out sets of 5 hours each for tuning and testing, all sampled at 16kHz.

The test sets are split into two development sets named "dev-clean" and "dev-other" and two test sets named "test-clean" and "test-other". The distinction between "clean" and "other" is based on the word error rate of a speech recognition system trained on the Wall Street Journal corpus [Paul & Baker 92, Panayotov & Chen$^+$ 15].

Detailed corpus statistics are presented in Table A.1.

Table A.1: Corpus data statistics for the LibriSpeech corpus.

partition	duration [h]	no. of words	OOV [%]
train	961.1	9.4M	-
dev-clean	5.4	54.4k	0.3
dev-other	5.1	50.9k	0.6
test-clean	5.4	52.6k	0.4
test-other	5.3	52.3k	0.5

A.2 Switchboard

The Switchboard corpus was released in 1992 by Texas Instruments Inc. [Godfrey & Holliman$^+$ 92]. It is a collection of narrow-band telephony conversations that were carried out by participants that were connected via a land-line telephony channel and given a topic to freely discuss. These conversations were recorded and later transcribed.

The training corpus contains 300h of speech and is published under catalog number LDC97S62 by the Linguistic Data Consortium (LDC). Results are reported on the Hub5'00 development set (LDC2002S09) and the Hub5'01 test set (LDC2002S13), which were collected under similar conditions.

Detailed corpus statistics are presented in Table A.2.

Table A.2: Training data statistics for the Switchboard corpus.

partition	duration [h]	no. of words	OOV [%]
Switchboard	311.7	3.1M	-
Hub5'00	3.6	40.6k	1.2
Hub5'01	6.2	59.7k	1.1

A.3 Quaero

Quaero was a French-German collaboration project to build a multimedia-based search engine which ran from 2008 to 2013 [Qua 08]. Within this project, speech recognition played a crucial role in indexing sections of media that contain human speech. The goal was to improve recognition accuracy on web-based content such as broadcast news, talk-shows and documentaries.

To measure the progress of speech recognition technologies, annual evaluation campaigns were held. During the first years no explicit training data was provided, but in 2010 and 2011 a total of 230 hours of transcribed broadcast conversations was released [Wiesler 17]. Out of the annual evaluation campaigns' test sets the dev'10 set is selected as development set and the latest test'13 set is used as the test set.

Detailed corpus statistics are presented in Table A.3.

Table A.3: Training data statistics for the Quaero corpus.

partition	duration [h]	no. of words	OOV [%]
train	232.4	2.8M	-
dev'10	3.3	39.0k	0.3
eval'13	3.1	34.2k	0.6

B. Derivations

B.1 Gradient of Maximum Mutual Information Criterion

We want to derive the gradient of the maximum mutual information training criterion for hybrid DNN-HMM based neural acoustic models of Equation 3.4.

$$\frac{\partial F_{\mathrm{MMI}}}{\partial \log p_\tau(a|x_1^T)} = \frac{\partial}{\partial \log p_\tau} \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \log \frac{p^\beta\left(w_1^N\right) \sum\limits_{\{(a_1^S, s_1^T) \in w_1^N\}} \prod\limits_{t=1}^{T} p_t^\alpha\left(a_{s_t} \middle| x_1^T\right)\ p^{-\gamma}(a_{s_t})\ p^\delta(s_t|s_{t-1})}{\sum\limits_{M, \{v_1^M\}} p^\beta(v_1^M) \sum\limits_{\{(a_1^S, s_1^T) \in v_1^M\}} \prod\limits_{t=1}^{T} p_t^\alpha(a_{s_t}|x_1^T)\ p^{-\gamma}(a_{s_t})\ p^\delta(s_t|s_{t-1})}$$

To ease the notation let the numerator be $\mathcal{N}$, the denominator be $\mathcal{D}$ and let ∇ denote the derivative.

$$\begin{aligned}
&= \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \nabla \log \frac{\mathcal{N}}{\mathcal{D}} \\
&= \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \nabla \left[\log \mathcal{N} - \log \mathcal{D}\right] \\
&= \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \left[\frac{\nabla \mathcal{N}}{\mathcal{N}} - \frac{\nabla \mathcal{D}}{\mathcal{D}}\right]
\end{aligned}$$

Now first consider the numerator part:

$$\frac{\nabla \mathcal{N}}{\mathcal{N}} = \frac{p^\beta\left(w_1^N\right) \sum\limits_{\{(a_1^S, s_1^T) \in w_1^N\}} \nabla \prod_{t=1}^{T} p_t^\alpha\left(a_{s_t} \middle| x_1^T\right)\ p^{-\gamma}(a_{s_t})\ p^\delta(s_t|s_{t-1})}{p^\beta(w_1^N) \sum\limits_{\{(a_1^S, s_1^T) \in w_1^N\}} \prod_{t=1}^{T} p_t^\alpha(a_{s_t}|x_1^T)\ p^{-\gamma}(a_{s_t})\ p^\delta(s_t|s_{t-1})}$$

with

$$\nabla p_t^\alpha\left(a_{s_t}\middle|x_1^T\right) = \frac{\partial p_t^\alpha\left(a_{s_t}\middle|x_1^T\right)}{\partial \log p_\tau(a|x_1^T)} = p_\tau\left(a\middle|x_1^T\right) \cdot \frac{\partial p_t^\alpha\left(a_{s_t}\middle|x_1^T\right)}{\partial p_\tau(a|x_1^T)} = \alpha p_t^\alpha\left(a_{s_t}\middle|x_1^T\right) \delta_{\tau,t}\delta_{a,a_{s_t}}$$

and since $\tau = t$ exactly once in $t = 1, ..., T$, the first Kronecker δ eliminates all other terms in the product rule of differentiation. The second Kronecker δ eliminates all state and label sequences that do not have the label a at position τ.

$$\begin{aligned}
&= \alpha \cdot \frac{p^\beta\left(w_1^N\right) \sum\limits_{\substack{\{(a_1^S,s_1^T)\in w_1^N\} \\ a_{s_\tau}=a}} \prod_{t=1}^T p_t^\alpha\left(a_{s_t}\middle|x_1^T\right)\, p^{-\gamma}(a_{s_t})\, p^\delta(s_t|s_{t-1})}{p^\beta(w_1^N) \sum\limits_{\{(a_1^S,s_1^T)\in w_1^N\}} \prod_{t=1}^T p_t^\alpha(a_{s_t}|x_1^T)\, p^{-\gamma}(a_{s_t})\, p^\delta(s_t|s_{t-1})} \\
&=: \alpha \cdot \gamma^N\left(a_{s_\tau} = a\middle|x_1^T, w_1^N\right)
\end{aligned}$$

Using the same steps, the denominator part becomes

$$\begin{aligned}
\frac{\nabla\mathcal{D}}{\mathcal{D}} &= \frac{\sum\limits_{M,\{v_1^M\}} p^\beta\left(v_1^M\right) \sum\limits_{\{(a_1^S,s_1^T)\in v_1^M\}} \nabla \prod_{t=1}^T p_t^\alpha\left(a_{s_t}\middle|x_1^T\right)\, p^{-\gamma}(a_{s_t})\, p^\delta(s_t|s_{t-1})}{\sum\limits_{M,\{v_1^M\}} p^\beta(v_1^M) \sum\limits_{\{(a_1^S,s_1^T)\in v_1^M\}} \prod_{t=1}^T p_t^\alpha(a_{s_t}|x_1^T)\, p^{-\gamma}(a_{s_t})\, p^\delta(s_t|s_{t-1})} \\
&= \alpha \cdot \frac{\sum\limits_{M,\{v_1^M\}} p^\beta\left(v_1^M\right) \sum\limits_{\substack{\{(a_1^S,s_1^T)\in v_1^M\} \\ a_{s_\tau}=a}} \prod_{t=1}^T p_t^\alpha\left(a_{s_t}\middle|x_1^T\right)\, p^{-\gamma}(a_{s_t})\, p^\delta(s_t|s_{t-1})}{\sum\limits_{M,\{v_1^M\}} p^\beta(v_1^M) \sum\limits_{\{(a_1^S,s_1^T)\in v_1^M\}} \prod_{t=1}^T p_t^\alpha(a_{s_t}|x_1^T)\, p^{-\gamma}(a_{s_t})\, p^\delta(s_t|s_{t-1})} \\
&=: \alpha \cdot \gamma^D\left(a_{s_\tau} = a\middle|x_1^T\right)
\end{aligned}$$

and therefore the complete gradient is

$$\frac{\partial F_{\text{MMI}}}{\partial \log p_\tau(a|x_1^T)} = \sum_{(w_1^N,x_1^T)\in\mathcal{T}} \alpha\left[\gamma^N\left(a_{s_\tau} = a\middle|x_1^T, w_1^N\right) - \gamma^D\left(a_{s_\tau} = a\middle|x_1^T\right)\right].$$

B.2 Gradient of State-Level Bayes Risk Criterion

We want to derive the gradient of the maximum mutual information training criterion for hybrid DNN-HMM based neural acoustic models of Equation 3.6. We loosely follow the derivation in [Povey 03].

$$\frac{\partial F_{\text{sMBR}}}{\partial \log p_\tau(a|x_1^T)} = \nabla \sum_{(w_1^N,x_1^T)\in\mathcal{T}} \frac{\sum\limits_{M,\{v_1^M\}} p^\beta\left(v_1^M\right) \sum\limits_{\{(a_1^S,s_1^T)\in v_1^M\}} \prod\limits_{t=1}^T p_t^\alpha\left(a_{s_t}\middle|x_1^T\right)\, p^{-\gamma}(a_{s_t})\, p^\delta(s_t|s_{t-1})\, A\left(a_1^T, \omega_1^T\right)}{\sum\limits_{M,\{v_1^M\}} p^\beta(v_1^M) \sum\limits_{\{(a_1^S,s_1^T)\in v_1^M\}} \prod\limits_{t=1}^T p_t^\alpha(a_{s_t}|x_1^T)\, p^{-\gamma}(a_{s_t})\, p^\delta(s_t|s_{t-1})}$$

To ease the notation let this numerator be $\mathcal{N}_a$, the denominator be $\mathcal{D}$ (same as in section B.1) and let ∇ denote the derivative.

$$
\begin{aligned}
&= \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \nabla \frac{\mathcal{N}_a}{\mathcal{D}} \\
&= \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \frac{[\nabla \mathcal{N}_a] \cdot \mathcal{D} - \mathcal{N}_a \cdot [\nabla \mathcal{D}]}{\mathcal{D}^2} \\
&= \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \frac{\nabla \mathcal{D}}{\mathcal{D}} \cdot \left[\frac{\nabla \mathcal{N}_a}{\nabla \mathcal{D}} - \frac{\mathcal{N}_a}{\mathcal{D}} \right]
\end{aligned}
$$

We already know $\nabla \mathcal{D}$ and $\frac{\nabla \mathcal{D}}{\mathcal{D}} = \gamma^D$ from the previous section, therefore consider now $\frac{\nabla \mathcal{N}_a}{\nabla \mathcal{D}}$

$$
\frac{\nabla \mathcal{N}_a}{\nabla \mathcal{D}} = \frac{\nabla \sum\limits_{M, \{v_1^M\}} p^\beta\left(v_1^M\right) \sum\limits_{\{(a_1^S, s_1^T) \in v_1^M\}} \prod\limits_{t=1}^{T} p_t^\alpha\left(a_{s_t} \middle| x_1^T\right)\, p^{-\gamma}(a_{s_t})\, p^\delta(s_t | s_{t-1}) \cdot A\left(a_1^T, \omega_1^T\right)}{\alpha \cdot \sum\limits_{M, \{v_1^M\}} p^\beta(v_1^M) \sum\limits_{\substack{\{(a_1^S, s_1^T) \in v_1^M\} \\ a_{s_\tau} = a}} \prod\limits_{t=1}^{T} p_t^\alpha(a_{s_t} | x_1^T)\, p^{-\gamma}(a_{s_t})\, p^\delta(s_t | s_{t-1})}
$$

using again the same steps as in $\nabla \mathcal{N}$ above

$$
\begin{aligned}
&= \frac{\sum\limits_{M, \{v_1^M\}} p^\beta\left(v_1^M\right) \sum\limits_{\substack{\{(a_1^S, s_1^T) \in v_1^M\} \\ a_{s_\tau} = a}} \prod_{t=1}^{T} p_t^\alpha\left(a_{s_t} \middle| x_1^T\right)\, p^{-\gamma}(a_{s_t})\, p^\delta(s_t | s_{t-1}) \cdot A\left(a_1^T, \omega_1^T\right)}{\sum\limits_{M, \{v_1^M\}} p^\beta(v_1^M) \sum\limits_{\substack{\{(a_1^S, s_1^T) \in v_1^M\} \\ a_{s_\tau} = a}} \prod_{t=1}^{T} p_t^\alpha(a_{s_t} | x_1^T)\, p^{-\gamma}(a_{s_t})\, p^\delta(s_t | s_{t-1})} \\
&=: \gamma^A\left(a_{s_\tau} = a \middle| x_1^T, \omega_1^T\right)
\end{aligned}
$$

Now γ^A is the average accuracy of all paths that contain the label a at position τ, whereas $\frac{\mathcal{N}_a}{\mathcal{D}}$ is the average accuracy $\bar{A}$ of the full utterance. Then the full gradient is

$$
\frac{\partial F_{\text{sMBR}}}{\partial \log p_\tau(a | x_1^T)} = \sum_{(w_1^N, x_1^T) \in \mathcal{T}} \alpha \cdot \gamma^D\left(a_{s_\tau} = a \middle| x_1^T\right) \cdot \left[\gamma^A\left(a_{s_\tau} = a \middle| x_1^T, \omega_1^T\right) - \bar{A} \right]
$$

C. Algorithms

C.1 TensorFlow Implementation of Approximative Recombination

The following listing contains the implementation of the approximative recombination procedure used in subsection 4.5.1. The function `custom_score_combine` should be added to the `ChoiceLayer` in the RETURNN network definition.

```
bsr_window_idx = 0
bsr_eos_token_idx = 0

bsr_allow_merge_of_padded = False
bsr_merge_eos_instantly = False  # bsr_allow_merge_of_padded will
                                 # have no effect if this is enabled


def custom_score_combine(layer, scores_in, scores_base, batch_dim,
                         scores_beam_in, base_beam_in, t, end_flags,
                         **kwargs):
    """
    :param ChoiceLayer layer: `ChoiceLayer` from which to fetch the
           `histories`
    :param tf.Tensor scores_in: (batch,base_beam_in,dim) the scores
           for each target word given the sequences in the beams so
           far (note: `dim` = `n_classes`)
    :param tf.Tensor scores_base: (batch,base_beam_in,dim|1) the
           score of each sequences within each beam within the batch
           until the current decoding step
    :param tf.Tensor batch_dim: () indicates the batch dimension size
           `batch`
    :param tf.Tensor scores_beam_in: () indicates the width of the
           beam to output (always beam_size in this case)
    :param tf.Tensor base_beam_in: () indicates the width of the base
           beam (initially 1 then beam_size)
    :param tf.Tensor t: () indicates the time step beginning at 0
    :param tf.Tensor end_flags: (batch,base_beam_in) indicates which
           sequences within the beam already terminated by setting
           the respective entry to 1
    :return: (batch,base_beam_in,dim) scores_base combined with
             scores_in according to the BSR algorithm
    :rtype: tf.Tensor
    """
    import tensorflow as tf

    ###################################################################
    #   Pre-process input
    ###################################################################

    cheating = layer.cheating
```

```
    assert cheating in ["exclusive", False], \
        f"Unsupported value for ChoiceLayer.cheating: '{cheating}'"

    histories = layer.explicit_search_sources[bsr_window_idx]. \
        output.copy_as_batch_major(). \
        placeholder  # (batch,beam,history)

    # Ensure that the histories are of shape (batch_dim,beam_size,
    # history_length). Note, if batch_dim is 1, then the batch
    # dimension is omitted, hence the insurance here. Note that in
    # the first search step, there are `beam_size` histories for
    # each batch entry. However, there is only 1 scores_base entry
    # for each batch entry. Note in this step applies
    # `base_beam_in` = 1 != `beam_size`, so there is a mismatch in
    # the beam dimension assumed by scores_base and the histories,
    # which are fetched from the WindowLayer. Since, in this step,
    # the histories all contain 0 anyway, the histories are cropped
    # by simply selecting the first history for each batch entry.
    histories = tf.reshape(
        histories,
        shape=[batch_dim, scores_beam_in, tf.shape(histories)[-1]]
    )
    histories = histories[:, 0:base_beam_in]

    scores_base = tf.squeeze(scores_base, axis=-1)  # (batch,beam,)
    scores_base_flat = tf.reshape(scores_base, [-1])  # (batch*beam,)

    ###########################################################
    #   Determine history matches
    ###########################################################

    # create a confusion matrix indicating the equality between the
    # histories within a beam
    h1 = tf.expand_dims(histories, axis=1)  # (batch,beam,1,history)
    h2 = tf.expand_dims(histories, axis=2)  # (batch,1,beam,history)
    confusion_matrix_word_matches = tf.equal(
        h1, h2
    )  # (batch,beam,beam,history)
    confusion_matrix_sequence_matches = tf.reduce_all(
        confusion_matrix_word_matches, axis=-1
    )  # (batch,beam,beam)

    # fetch the index tuples (batch, beam_i, beam_j) for the entries
    # of the confusion matrix, which indicate history matches. The
    # output are these tuples (batch, beam_i, beam_j) where beam_i
    # and beam_j have the same predecessor words
    tmp_indices = tf.cast(
        tf.where(confusion_matrix_sequence_matches), dtype=tf.int32
    )  # (?,3)

    # This delivers for all recent histories within the batch, the
    # index of the first matching recent history within its
    # respective beam, regardless of if this is the recent history
    # itself, or a recent history from a different sequence,
    # appearing earlier in the sequence order. Note that the recent
    # histories are expected to be ordered by their sequence's score,
    # meaning that the index of the matching recent history of the
    # sequence with the highest sequence score is obtained
    # automatically.
    segment_data = tmp_indices[:, -1]
    segment_ids = base_beam_in * tmp_indices[:, 0] + tmp_indices[:,1]
```

```
    match_indices = tf.segment_min(segment_data, segment_ids)
    match_indices.set_shape(scores_base_flat.shape)  # (batch*beam,)

    ###################################################################
    #   if merge eos instantly
    ###################################################################

    if bsr_merge_eos_instantly:
        # If `bsr_merge_eos_instantly` is set, the terminated
        # sequences should be merged immediately, disregarding their
        # BSR histories. If it is not set, terminated sequences would
        # be merged only if they share the same BSR history, which
        # becomes (mainly) the case when the history window covers
        # just the padding eos-tokens of those sequences.

        # Create a mask to "excluded" non-eos positions from the
        # subsequent reduce_min operation. This is done by simply
        # setting all excluded entries to a sufficiently high value
        not_end_flags = tf.ones_like(end_flags, dtype=tf.int32) \
                        - tf.cast(end_flags, tf.int32)
        not_end_mask = not_end_flags \
                       * (tf.shape(match_indices)[0] + 1)
        # Determine the beam index into which to merge all terminated
        # sequences within a beam
        eos_matches = tf.reshape(
            tf.range(tf.shape(match_indices)[0]) % base_beam_in,
            shape=tf.shape(end_flags)
        ) * tf.cast(end_flags, tf.int32) + not_end_mask
        eos_matches = tf.reduce_min(
            eos_matches, axis=-1, keep_dims=True
        )
        # Alter the match_indices such that the terminated sequences
        # are merged into the correct entry
        eos_mask = tf.cast(end_flags, tf.int32) * eos_matches
        match_indices = match_indices * tf.reshape(
            not_end_flags, shape=tf.shape(match_indices)
        ) + tf.reshape(eos_mask, shape=tf.shape(match_indices))

    ###################################################################
    #   if reference in beam
    ###################################################################

    if cheating == "exclusive":
        # If cheating is 'exclusive', the reference target sequence
        # will be always in the last entry of the respective beam.
        # If the reference is about to get recombined with one or
        # more sequences within the beam, it is ensured that those
        # sequences are merged into the reference regardless of
        # whether it has the highest score among these sequences.

        # Fetch the indices of the beam entries into which the
        # reference are about to get merged
        reference_merge_target = tf.expand_dims(
            match_indices[base_beam_in - 1::base_beam_in], -1
        )  # (batch,1)
        # Now search the beams for other sequences, which are about
        # to get merged into these entries
        reference_merge_siblings = tf.reshape(
            match_indices, shape=tf.shape(scores_base)
        )  # (batch,beam)
        reference_merge_siblings = tf.cast(
```

```
            tf.equal(reference_merge_siblings,
                     reference_merge_target),
            dtype=tf.int32
        )  # (batch,beam)
        reference_merge_siblings = tf.reshape(
            reference_merge_siblings,
            shape=tf.shape(match_indices)
        )  # (batch*beam,)
        not_reference_merge_siblings = tf.ones_like(
            reference_merge_siblings,
            dtype=tf.int32
        ) - reference_merge_siblings

        # Redirect all affected sequences to the reference position
        match_indices = match_indices * not_reference_merge_siblings
        match_indices = match_indices + reference_merge_siblings \
                        * (base_beam_in - 1)

    ####################################################################
    #   Filter padded sequences and terminated searches
    ####################################################################

    # Create masks, indicating which sequences are allowed to be
    # recombined and which aren't (or the opposite), according to
    # the rules described at `is_padded` and `is_search_terminated`
    # Note: the logical 'or' (= '+') is not used, because values
    # being 0 or 1 are required, and 1 + 1 = 2.
    merge_allow_mask = tf.ones_like(scores_base, dtype=tf.int32)

    # If bsr_allow_merge_of_padded or bsr_merge_eos_instantly, the
    # subsequent mask is not applied...
    if not bsr_allow_merge_of_padded and not bsr_merge_eos_instantly:
        # Check for each sequence in the beam, whether it is padded
        # (has more than 1 tailing </s>). For sequences with a
        # tailing </s> only those should to be recombined, which
        # just terminated, not those which were already padded with
        # extra </s> tokens.
        is_padded = tf.cast(
            tf.equal(tf.reduce_sum(histories[:, :, -2:], axis=-1),
                     bsr_eos_token_idx),
            tf.int32
        )
        is_not_padded = tf.ones_like(is_padded) - is_padded
        merge_allow_mask = merge_allow_mask * is_not_padded

    # Check for each beam in the batch, whether the search is
    # already terminated (all end_flags = 1). In case the search is
    # already terminated, the sequences should not to be recombined
    # no matter which histories they have, because there would be no
    # sequence to take their place in the beam.
    is_search_terminated = tf.reduce_prod(
        tf.cast(end_flags, tf.int32), axis=-1
    )
    is_not_search_terminated = tf.expand_dims(
        tf.ones_like(is_search_terminated, dtype=tf.int32)
        - is_search_terminated, axis=-1
    )
    merge_allow_mask = merge_allow_mask * is_not_search_terminated

    # finalize the merge masks
    merge_allow_mask = tf.reshape(
```

```
        merge_allow_mask, tf.shape(match_indices)
    )
    merge_forbid_mask = tf.ones_like(merge_allow_mask) \
                        - merge_allow_mask

    # Using these masks, (re-)direct those sequences, which are not
    # allowed to be merged into another sequence, to their original
    # index
    match_indices = match_indices * merge_allow_mask
    match_indices = match_indices \
                    + (tf.range(tf.shape(match_indices)[0])
                       % base_beam_in) \
                    * merge_forbid_mask

    ###################################################################
    #   Determine recombinations
    ###################################################################

    # Next, construct a list of scatter indices, containing for each
    # sequence within the batch tensor:
    # * the batch index `batch_idx`, indicating the index of the
    #   originating input sequence within the batch
    # * the match index `match_idx`, indicating the index of the
    #   first sequence with a matching history, respective to the
    #   beam
    # * the stack index `stack_idx`, indicating the originating
    #   index of the sequence, respective to its beam
    batch_idx = tf.range(tf.shape(match_indices)[0]) \
                // base_beam_in  # (batch*beam,)
    match_idx = match_indices  # (batch*beam,)
    stack_idx = tf.range(tf.shape(match_indices)[0]) \
                % base_beam_in  # (batch*beam,)
    # E.g. the columns result in the the scatter indices:
    # [0 0 0 0  1 1 1 1  2 2 2 2] = batch_idx
    # [0 1 0 3  0 1 1 3  0 1 2 2] = match_idx
    # [0 1 2 3  0 1 2 3  0 1 2 3] = stack_idx
    scatter_indices = tf.stack(
        [batch_idx, match_idx, stack_idx], axis=-1
    )  # (batch*beam,2)

    ###################################################################
    #   Perform recombinations and compute scores
    ###################################################################

    # Now, scores_base should be combined, which are (pseudo)
    # log-probs. This is done by summing them in probability space.
    # To compute this efficiently, the LogSumExp trick is used:

    # To apply LogSumExp, the scatter_indices and scatter_nd are
    # used, to create a scatter_stack matrix, containing the scores
    # which are meant to be combined with LogSumExp under its last
    # dimension. However, those matrices will have entries, which
    # were initialized with 0, and which should be excluded by
    # LogSumExp. To exclude them, those entries are masked out by
    # setting them to -inf (= log(0)), hence LogSumExp will treat
    # those entries as 0 in its internal sum. The following tensors
    # are used to create that -inf mask:
    scatter_track = tf.scatter_nd(
        scatter_indices,
        tf.ones_like(scores_base_flat, dtype=tf.int32),
        tf.shape(confusion_matrix_sequence_matches)
```

```
    )  # (batch,beam,beam)
    neg_inf_tensor = float('-inf') \
                     * tf.ones_like(scatter_track, dtype=tf.float32)
                     # (batch,beam,beam)
    neg_inf_mask = tf.equal(scatter_track, 0)  # (batch,beam,beam)

    # create the scatter_stack tensor by scattering the scores_base
    # to its designated positions and apply the -inf mask to unused
    # entries:
    scatter_stack = tf.scatter_nd(
        scatter_indices, scores_base_flat, tf.shape(
            confusion_matrix_sequence_matches
        )
    )  # (batch,beam,beam)
    scatter_stack = tf.where(
        neg_inf_mask, neg_inf_tensor, scatter_stack
    )
    # now combine the scores by applying LogSumExp:
    new_scores_base = tf.reduce_logsumexp(
        scatter_stack, axis=-1, keep_dims=True
    )  # (batch,beam,dim|1)
    scores_out = new_scores_base + scores_in
    return scores_out
```

Listing C.1: Python `custom_score_combine` function used for the RETURNN implementation of the approximative recombination algorithm. The function identifies entries in the search beam that share a common history and combines the probability values. There are special treatments for the reference word sequence and sequences that are complete, i.e. end in the sentence-end token. Taken from [Wynands 21].

List of Figures

3.1 Frame-MMI Probability Mass Distribution 20

4.1 Trellis Example 27

List of Tables

3.1 Default Transition Probabilities . . . 13
3.2 Frame-MMI: Average Entropy . . . 16
3.3 Frame-MMI: Interpolation Factor . . . 16
3.4 Hybrid WER Overview . . . 17
3.5 Literature Overview: Sequence Training Hybrid Models . . . 21

4.1 Unlimited Context Models: Trellis Statistics . . . 30
4.2 Unlimited Context Models: Search Space WER comparison . . . 31
4.3 Limited Context Models: Trellis Statistics . . . 32
4.4 Limited Context Models: Search Space WER comparison . . . 32
4.5 Approximative Recombination Overview . . . 33
4.6 Language Model Tuning: Context Length . . . 33
4.7 Language Model Tuning: Overfitting . . . 34
4.8 Language Model Tuning: Training Data . . . 34
4.9 Attention Sequence Training Criteria Comparison . . . 35
4.10 Literature Overview: Sequence Training End-To-End Models . . . 39

5.1 Automatic Scale Learning: Symbol-Agnostic Scales . . . 44
5.2 Automatic Scale Learning: Symbol-Dependent Scales . . . 45
5.3 Automatic Scale Learning: Joint Training . . . 46
5.4 Decision Rule Comparison . . . 47
5.5 Language Model Exchange . . . 48
5.6 Training Criteria: Speed Comparison . . . 49
5.7 Training Criteria: WER Comparison . . . 49
5.8 Literature Overview: Symbol-Level LM Integration for End-To-End Models . . . 54
5.9 Literature Overview: Symbol-Dependent Scaling Factors . . . 55

6.1 ILM Estimation Method PPL/WER Comparison . . . 61
6.2 ILM PPL Comparison After LM Integration . . . 62
6.3 ILM WER Comparison . . . 62
6.4 Literature Overview: ILM Correction for End-To-End Models . . . 67

A.1 Corpus Data Statistics: LibriSpeech . . . 71
A.2 Corpus Data Statistics: Switchboard . . . 72
A.3 Corpus Data Statistics: Quaero . . . 72

Bibliography

[Abadi & Barham+ 16] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D.G. Murray, B. Steiner, P.A. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng: "TensorFlow: A System for Large-Scale Machine Learning." In *12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016, Savannah, GA, USA, November 2-4, 2016*, pp. 265–283. USENIX Association, 2016.

[Anil & Dai+ 23] R. Anil, A.M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen, E. Chu, J.H. Clark, L.E. Shafey, Y. Huang, K. Meier-Hellstern, G. Mishra, E. Moreira, M. Omernick, K. Robinson, S. Ruder, Y. Tay, K. Xiao, Y. Xu, Y. Zhang, G.H. Ábrego, J. Ahn, J. Austin, P. Barham, J.A. Botha, J. Bradbury, S. Brahma, K. Brooks, M. Catasta, Y. Cheng, C. Cherry, C.A. Choquette-Choo, A. Chowdhery, C. Crepy, S. Dave, M. Dehghani, S. Dev, J. Devlin, M. Díaz, N. Du, E. Dyer, V. Feinberg, F. Feng, V. Fienber, M. Freitag, X. Garcia, S. Gehrmann, L. Gonzalez, et al.: "PaLM 2 Technical Report." arXiv:2305.10403, 2023.

[Bahdanau & Cho+ 15] D. Bahdanau, K. Cho, Y. Bengio: "Neural Machine Translation by Jointly Learning to Align and Translate." In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[Bahl & Brown+ 86] L.R. Bahl, P.F. Brown, P.V. de Souza, R.L. Mercer: "Maximum mutual information estimation of hidden Markov model parameters for speech recognition." In *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 1986, Tokyo, Japan, April 7-11, 1986*, pp. 49–52. IEEE, 1986.

[Bahl & Brown+ 88] L.R. Bahl, P.F. Brown, P.V. de Souza, R.L. Mercer: "A new algorithm for the estimation of hidden Markov model parameters." In *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '88, New York, New York, USA, April 11-14, 1988*, pp. 493–497. IEEE, 1988.

[Bai & Yi+ 19] Y. Bai, J. Yi, J. Tao, Z. Tian, Z. Wen: "Learn Spelling from Teachers: Transferring Knowledge from Language Models to Sequence-to-Sequence Speech Recognition." In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pp. 3795–3799. ISCA, 2019.

[Bai & Yi+ 21] Y. Bai, J. Yi, J. Tao, Z. Wen, Z. Tian, S. Zhang: "Integrating Knowledge Into End-to-End Speech Recognition From External Text-Only Data." *IEEE ACM Trans. Audio Speech Lang. Process.*, Vol. 29, pp. 1340–1351, 2021.

[Baum & Petrie 66] L.E. Baum, T. Petrie: "Statistical inference for probabilistic functions of finite state Markov chains." *The annals of mathematical statistics*, Vol. 37, No. 6, pp. 1554–1563, 1966.

[Bayes 63] T. Bayes: "An essay towards solving a problem in the doctrine of chances." *Phil. Trans. of the Royal Soc. of London*, Vol. 53, pp. 370–418, 1763.

[Beck & Zhou+ 19] E. Beck, W. Zhou, R. Schlüter, H. Ney: "LSTM Language Models for LVCSR in First-Pass Decoding and Lattice-Rescoring." arXiv:1907.01030, 2019.

[Bergstra & Breuleux+ 10] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J.P. Turian, D. Warde-Farley, Y. Bengio: "Theano: A CPU and GPU Math Compiler in Python." In *Proceedings of the 9th Python in Science Conference 2010 (SciPy 2010), Austin, Texas, June 28 - July 3, 2010*, pp. 18–24. scipy.org, 2010.

[Cabrera & Liu+ 21] R. Cabrera, X. Liu, M. Ghodsi, Z. Matteson, E. Weinstein, A. Kannan: "Language model fusion for streaming end to end speech recognition." arXiv:2104.04487, 2021.

[CC4 13] "Creative Commons Attribution 4.0 License", 2013. `https://creativecommons.org/licenses/by/4.0/`.

[Chang & Lahiri+ 15] S. Chang, A. Lahiri, I. Alphonso, B. Oguz, M. Levit, B. Dumoulin: "Discriminative training of context-dependent language model scaling factors and interpolation weights." In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2015, Scottsdale, AZ, USA, December 13-17, 2015*, pp. 45–51. IEEE, 2015.

[Chiu & Sainath+ 18] C. Chiu, T.N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R.J. Weiss, K. Rao, E. Gonina, N. Jaitly, B. Li, J. Chorowski, M. Bacchiani: "State-of-the-Art Speech Recognition with Sequence-to-Sequence Models." In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018*, pp. 4774–4778. IEEE, 2018.

[Cho & Watanabe+ 19] J. Cho, S. Watanabe, T. Hori, M.K. Baskar, H. Inaguma, J. Villalba, N. Dehak: "Language Model Integration Based on Memory Control for Sequence to Sequence Speech Recognition." In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pp. 6191–6195. IEEE, 2019.

[Chou & Lee+ 93] W. Chou, C. Lee, B. Juang: "Minimum error rate training based on N-best string models." In *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '93, Minneapolis, Minnesota, USA, April 27-30, 1993*, pp. 652–655. IEEE Computer Society, 1993.

[Das & Sunkara+ 23] N. Das, M. Sunkara, S. Bodapati, J. Cai, D. Kulshreshtha, J. Farris, K. Kirchhoff: "Mask the Bias: Improving Domain-Adaptive Generalization of CTC-Based ASR with Internal Language Model Estimation." In *IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP 2023, Rhodes Island, Greece, June 4-10, 2023*, pp. 1–5. IEEE, 2023.

[Doetsch & Zeyer+ 17] P. Doetsch, A. Zeyer, P. Voigtlaender, I. Kulikov, R. Schlüter, H. Ney: "Returnn: The RWTH extensible training framework for universal recurrent neural networks." In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*, pp. 5345–5349. IEEE, 2017.

[Gage 94] P. Gage: "A new algorithm for data compression." *C Users J.*, Vol. 12, No. 2, pp. 23–38, feb 1994.

[Gibson & Hain 06] M. Gibson, T. Hain: "Hypothesis spaces for minimum Bayes risk training in large vocabulary speech recognition." In *INTERSPEECH 2006 - ICSLP, Ninth International Conference on Spoken Language Processing, Pittsburgh, PA, USA, September 17-21, 2006*. ISCA, 2006.

[Godfrey & Holliman+ 92] J.J. Godfrey, E. Holliman, J. McDaniel: "SWITCHBOARD: telephone speech corpus for research and development." In *1992 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '92, San Francisco, California, USA, March 23-26, 1992*, pp. 517–520. IEEE Computer Society, 1992.

[Goodfellow & Warde-Farley+ 13] I.J. Goodfellow, D. Warde-Farley, M. Mirza, A.C. Courville, Y. Bengio: "Maxout Networks." In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, Vol. 28 of *JMLR Workshop and Conference Proceedings*, pp. 1319–1327. JMLR.org, 2013.

[Goronzy & Rapp+ 04] S. Goronzy, S. Rapp, R. Kompe: "Generating non-native pronunciation variants for lexicon adaptation." *Speech Commun.*, Vol. 42, No. 1, pp. 109–123, 2004.

[Graves & Fernández+ 06] A. Graves, S. Fernández, F.J. Gomez, J. Schmidhuber: "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks." In W.W. Cohen, A.W. Moore, editors, *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, Vol. 148 of *ACM International Conference Proceeding Series*, pp. 369–376. ACM, 2006.

[Graves & Jaitly+ 13] A. Graves, N. Jaitly, A. Mohamed: "Hybrid speech recognition with Deep Bidirectional LSTM." In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding, Olomouc, Czech Republic, December 8-12, 2013*, pp. 273–278. IEEE, 2013.

[Gülçehre & Firat+ 15] Ç. Gülçehre, O. Firat, K. Xu, K. Cho, L. Barrault, H. Lin, F. Bougares, H. Schwenk, Y. Bengio: "On Using Monolingual Corpora in Neural Machine Translation." arXiv:1503.03535, 2015.

[Guo & Tiwari+ 20] J. Guo, G. Tiwari, J. Droppo, M.V. Segbroeck, C. Huang, A. Stolcke, R. Maas: "Efficient Minimum Word Error Rate Training of RNN-Transducer for End-to-End Speech Recognition." In *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, pp. 2807–2811. ISCA, 2020.

[Hinton & Vinyals+ 15] G.E. Hinton, O. Vinyals, J. Dean: "Distilling the Knowledge in a Neural Network." arXiv:1503.02531, 2015.

[Hochreiter & Schmidhuber 97] S. Hochreiter, J. Schmidhuber: "Long Short-Term Memory." *Neural Comput.*, Vol. 9, No. 8, pp. 1735–1780, 1997.

[Hoffmeister & Liang+ 09] B. Hoffmeister, R. Liang, R. Schlüter, H. Ney: "Log-linear model combination with word-dependent scaling factors." In *INTERSPEECH 2009, 10th Annual Conference of the International Speech Communication Association, Brighton, United Kingdom, September 6-10, 2009*, pp. 248–251. ISCA, 2009.

[Hu & Cai+ 17] W. Hu, M. Cai, K. Chen, H. Ding, L. Sun, S. Liang, X. Mo, Q. Huo: "Sequence Discriminative Training for Offline Handwriting Recognition by an Interpolated CTC

and Lattice-Free MMI Objective Function." In *14th IAPR International Conference on Document Analysis and Recognition, ICDAR 2017, Kyoto, Japan, November 9-15, 2017*, pp. 61–66. IEEE, 2017.

[Huang & Li+ 14] Z. Huang, J. Li, C. Weng, C. Lee: "Beyond cross-entropy: towards better frame-level objective functions for deep neural network training in automatic speech recognition." In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pp. 1214–1218. ISCA, 2014.

[Kanda & Meng+ 21] N. Kanda, Z. Meng, L. Lu, Y. Gaur, X. Wang, Z. Chen, T. Yoshioka: "Minimum Bayes Risk Training for End-to-End Speaker-Attributed ASR." In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pp. 6503–6507. IEEE, 2021.

[Kim & Shangguan+ 21] S. Kim, Y. Shangguan, J. Mahadeokar, A. Bruguier, C. Fuegen, M.L. Seltzer, D. Le: "Improved Neural Language Model Fusion for Streaming Recurrent Neural Network Transducer." In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pp. 7333–7337. IEEE, 2021.

[Lee & Chang 22] C.W. Lee, J.H. Chang: "Improving transformer-based acoustic model performance using sequence discriminative training." *The Journal of the Acoustical Society of Korea*, Vol. 41, No. 3, pp. 335–341, 2022.

[Liu & Ma+ 22] Y. Liu, R. Ma, H. Xu, Y. He, Z. Ma, W. Zhang: "Internal Language Model Estimation Through Explicit Context Vector Learning for Attention-based Encoder-decoder ASR." In *Interspeech 2022, 23rd Annual Conference of the International Speech Communication Association, Incheon, Korea, 18-22 September 2022*, pp. 1666–1670. ISCA, 2022.

[Lu & Meng+ 21] L. Lu, Z. Meng, N. Kanda, J. Li, Y. Gong: "On Minimum Word Error Rate Training of the Hybrid Autoregressive Transducer." In *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia, 30 August - 3 September 2021*, pp. 3435–3439. ISCA, 2021.

[Lu & Xiao+ 19] L. Lu, X. Xiao, Z. Chen, Y. Gong: "PyKaldi2: Yet another speech toolkit based on Kaldi and PyTorch." arXiv:1907.05955, 2019.

[Lüscher & Beck+ 19] C. Lüscher, E. Beck, K. Irie, M. Kitza, W. Michel, A. Zeyer, R. Schlüter, H. Ney: "RWTH ASR Systems for LibriSpeech: Hybrid vs Attention." In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pp. 231–235. ISCA, 2019.

[Lyu & Tan+ 10] D. Lyu, T.P. Tan, E. Chng, H. Li: "SEAME: a Mandarin-English code-switching speech corpus in south-east asia." In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pp. 1986–1989. ISCA, 2010.

[Mann & Raissi+ 23] D. Mann, T. Raissi, W. Michel, R. Schlüter, H. Ney: "End-To-End Training of a Neural HMM with Label and Transition Probabilities." In *IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2023, Taipei, Taiwan, December 16-20, 2023*, pp. 1–8. IEEE, 2023.

[Manohar & Povey+ 15] V. Manohar, D. Povey, S. Khudanpur: "Semi-supervised maximum mutual information training of deep neural network acoustic models." In *INTERSPEECH 2015, 16th Annual Conference of the International Speech Communication Association, Dresden, Germany, September 6-10, 2015*, pp. 2630–2634. ISCA, 2015.

[McDermott & Sak+ 19] E. McDermott, H. Sak, E. Variani: "A Density Ratio Approach to Language Model Fusion in End-to-End Automatic Speech Recognition." In *IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2019, Singapore, December 14-18, 2019*, pp. 434–441. IEEE, 2019.

[McGuire 05] H. McGuire: "LibriVox Website", 2005. `https://librivox.org/`.

[Meng & Kanda+ 21] Z. Meng, N. Kanda, Y. Gaur, S. Parthasarathy, E. Sun, L. Lu, X. Chen, J. Li, Y. Gong: "Internal Language Model Training for Domain-Adaptive End-To-End Speech Recognition." In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pp. 7338–7342. IEEE, 2021.

[Meng & Parthasarathy+ 21] Z. Meng, S. Parthasarathy, E. Sun, Y. Gaur, N. Kanda, L. Lu, X. Chen, R. Zhao, J. Li, Y. Gong: "Internal Language Model Estimation for Domain-Adaptive End-to-End Speech Recognition." In *IEEE Spoken Language Technology Workshop, SLT 2021, Shenzhen, China, January 19-22, 2021*, pp. 243–250. IEEE, 2021.

[Meng & Wu+ 21] Z. Meng, Y. Wu, N. Kanda, L. Lu, X. Chen, G. Ye, E. Sun, J. Li, Y. Gong: "Minimum Word Error Rate Training with Language Model Fusion for End-to-End Speech Recognition." In *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia, 30 August - 3 September 2021*, pp. 2596–2600. ISCA, 2021.

[Meyer & Michel+ 22] F. Meyer, W. Michel, M. Zeineldeen, R. Schlüter, H. Ney: "Automatic Learning of Subword Dependent Model Scales." In *Interspeech 2022, 23rd Annual Conference of the International Speech Communication Association, Incheon, Korea, 18-22 September 2022*, pp. 4133–4136. ISCA, 2022.

[Michel & Schlüter+ 20a] W. Michel, R. Schlüter, H. Ney: "Early Stage LM Integration Using Local and Global Log-Linear Combination." In *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, pp. 3605–3609. ISCA, 2020.

[Michel & Schlüter+ 20b] W. Michel, R. Schlüter, H. Ney: "Frame-Level MMI as A Sequence Discriminative Training Criterion for LVCSR." In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pp. 6904–6908. IEEE, 2020.

[Miclet & Mercier 88] L. Miclet, G. Mercier: "Evaluation of the acoustic decoder of the 'KEAL' speech recognition system." In *9th International Conference on Pattern Recognition, ICPR 1988, 14-17 November 1988, Ergife Palace Hotel, Rome, Italy*, pp. 481–483. IEEE, 1988.

[Morgan & Bourlard+ 93] N. Morgan, H. Bourlard, S. Renals, M. Cohen, H. Franco: "Hybrid Neural Network/Hidden Markov Model Systems for Continuous Speech Recognition." *Int. J. Pattern Recognit. Artif. Intell.*, Vol. 7, No. 4, pp. 899–916, 1993.

[Panayotov & Chen+ 15] V. Panayotov, G. Chen, D. Povey, S. Khudanpur: "Librispeech: An ASR corpus based on public domain audio books." In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2015, South Brisbane, Queensland, Australia, April 19-24, 2015*, pp. 5206–5210. IEEE, 2015.

[Park & Chan+ 19] D.S. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E.D. Cubuk, Q.V. Le: "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition." In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pp. 2613–2617. ISCA, 2019.

[Paszke & Gross+ 19] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E.Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala: "PyTorch: An Imperative Style, High-Performance Deep Learning Library." arXiv:1912.01703, 2019.

[Paul & Baker 92] D.B. Paul, J.M. Baker: "The design for the wall street journal-based CSR corpus." In *The Second International Conference on Spoken Language Processing, ICSLP 1992, Banff, Alberta, Canada, October 13-16, 1992*, pp. 899–902. ISCA, 1992.

[Peng & Liu+ 22] Y. Peng, Y. Liu, J. Zhang, H. Xu, Y. He, H. Huang, E.S. Chng: "Internal Language Model Estimation based Language Model Fusion for Cross-Domain Code-Switching Speech Recognition." arXiv:2207.04176, 2022.

[Peter & Beck+ 18] J. Peter, E. Beck, H. Ney: "Sisyphus, a Workflow Manager Designed for Machine Translation and Automatic Speech Recognition." In E. Blanco, W. Lu, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pp. 84–89. Association for Computational Linguistics, 2018.

[Povey 03] D. Povey: "Discriminative Training for Large Vocabulary Speech Recognition." Ph.D. thesis, University of Cambridge, UK, 2003.

[Povey & Peddinti+ 16] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, S. Khudanpur: "Purely Sequence-Trained Neural Networks for ASR Based on Lattice-Free MMI." In *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*, pp. 2751–2755. ISCA, 2016.

[Povey & Woodland 02] D. Povey, P.C. Woodland: "Minimum Phone Error and I-smoothing for improved discriminative training." In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2002, May 13-17 2002, Orlando, Florida, USA*, pp. 105–108. IEEE, 2002.

[Prabhavalkar & Hori+ 24] R. Prabhavalkar, T. Hori, T.N. Sainath, R. Schlüter, S. Watanabe: "End-to-End Speech Recognition: A Survey." *IEEE ACM Trans. Audio Speech Lang. Process.*, Vol. 32, pp. 325–351, 2024.

[Prabhavalkar & Sainath+ 18] R. Prabhavalkar, T.N. Sainath, Y. Wu, P. Nguyen, Z. Chen, C. Chiu, A. Kannan: "Minimum Word Error Rate Training for Attention-Based Sequence-to-Sequence Models." In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018*, pp. 4839–4843. IEEE, 2018.

[Qua 08] "Quaero Website", 2008. `http://www.quaero.org/`.

[Rossenbach & Zeyer+ 20] N. Rossenbach, A. Zeyer, R. Schlüter, H. Ney: "Generating Synthetic Audio Data for Attention-Based Speech Recognition Systems." In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pp. 7069–7073. IEEE, 2020.

[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 2009, 10th Annual Conference of the International Speech Communication Association, Brighton, United Kingdom, September 6-10, 2009*, pp. 2111–2114. ISCA, 2009.

[Sak & Senior+ 15] H. Sak, A.W. Senior, K. Rao, F. Beaufays: "Fast and accurate recurrent neural network acoustic models for speech recognition." In *INTERSPEECH 2015, 16th Annual Conference of the International Speech Communication Association, Dresden, Germany, September 6-10, 2015*, pp. 1468–1472. ISCA, 2015.

[Saon & Sercu+ 16] G. Saon, T. Sercu, S.J. Rennie, H.J. Kuo: "The IBM 2016 English Conversational Telephone Speech Recognition System." In *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*, pp. 7–11. ISCA, 2016.

[Schlüter & Bezrukov+ 07] R. Schlüter, I. Bezrukov, H. Wagner, H. Ney: "Gammatone Features and Feature Combination for Large Vocabulary Speech Recognition." In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2007, Honolulu, Hawaii, USA, April 15-20, 2007*, pp. 649–652. IEEE, 2007.

[Schlüter & Müller+ 99] R. Schlüter, B. Müller, F. Wessel, H. Ney: "Interdependence of Language Models and Discriminative Training." In *IEEE Automatic Speech Recognition and Understanding Workshop*, Vol. 1, pp. 119–122, Keystone, CO, Dec. 1999.

[Schlüter & Ney 01] R. Schlüter, H. Ney: "Model-based MCE bound to the true Bayes' error." *IEEE Signal Process. Lett.*, Vol. 8, No. 5, pp. 131–133, 2001.

[Shan & Weng+ 19] C. Shan, C. Weng, G. Wang, D. Su, M. Luo, D. Yu, L. Xie: "Component Fusion: Learning Replaceable Language Model Component for End-to-end Speech Recognition System." In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pp. 5631–5635. IEEE, 2019.

[Shannon 17] M. Shannon: "Optimizing Expected Word Error Rate via Sampling for Speech Recognition." In *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017*, pp. 3537–3541. ISCA, 2017.

[Shi & Feng+ 20] X. Shi, Q. Feng, L. Xie: "The ASRU 2019 Mandarin-English Code-Switching Speech Recognition Challenge: Open Datasets, Tracks, Methods and Results." arXiv:2007.05916, 2020.

[Sriram & Jun+ 18] A. Sriram, H. Jun, S. Satheesh, A. Coates: "Cold Fusion: Training Seq2Seq Models Together with Language Models." In *Interspeech 2018, 19th Annual Conference of the International Speech Communication Association, Hyderabad, India, 2-6 September 2018*, pp. 387–391. ISCA, 2018.

[Srivastava & Hinton+ 14] N. Srivastava, G.E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov: "Dropout: a simple way to prevent neural networks from overfitting." *J. Mach. Learn. Res.*, Vol. 15, No. 1, pp. 1929–1958, 2014.

[Stahlberg & Cross+ 18] F. Stahlberg, J. Cross, V. Stoyanov: "Simple Fusion: Return of the Language Model." In *Proceedings of the Third Conference on Machine Translation: Research Papers, WMT 2018, Belgium, Brussels, October 31 - November 1, 2018*, pp. 204–211. Association for Computational Linguistics, 2018.

[Su & Li+ 13] H. Su, G. Li, D. Yu, F. Seide: "Error back propagation for sequence training of Context-Dependent Deep NetworkS for conversational speech transcription." In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pp. 6664–6668. IEEE, 2013.

[Tachioka & Watanabe+ 14] Y. Tachioka, S. Watanabe, J.L. Roux, J.R. Hershey: "Sequence discriminative training for low-rank deep neural networks." In *2014 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2014, Atlanta, GA, USA, December 3-5, 2014*, pp. 572–576. IEEE, 2014.

[Tian & Yu+ 22] J. Tian, J. Yu, C. Weng, S. Zhang, D. Su, D. Yu, Y. Zou: "Consistent Training and Decoding for End-to-End Speech Recognition Using Lattice-Free MMI." In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022*, pp. 7782–7786. IEEE, 2022.

[Tian & Yu+ 23] J. Tian, J. Yu, C. Weng, Y. Zou, D. Yu: "Integrating Lattice-Free MMI Into End-to-End Speech Recognition." *IEEE ACM Trans. Audio Speech Lang. Process.*, Vol. 31, pp. 25–38, 2023.

[Tian & Zhang+ 17] X. Tian, J. Zhang, Z. Ma, Y. He, J. Wei, P. Wu, W. Situ, S. Li, Y. Zhang: "Deep LSTM for Large Vocabulary Continuous Speech Recognition." arXiv:1703.07090, 2017.

[Toshniwal & Kannan+ 18] S. Toshniwal, A. Kannan, C. Chiu, Y. Wu, T.N. Sainath, K. Livescu: "A Comparison of Techniques for Language Model Integration in Encoder-Decoder Speech Recognition." In *2018 IEEE Spoken Language Technology Workshop, SLT 2018, Athens, Greece, December 18-21, 2018*, pp. 369–375. IEEE, 2018.

[Touvron & Lavril+ 23] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, G. Lample: "LLaMA: Open and Efficient Foundation Language Models." arXiv:2302.13971, 2023.

[Variani & Riley+ 22] E. Variani, M. Riley, D. Rybach, C. Allauzen, T. Chen, B. Ramabhadran: "On Adaptive Weight Interpolation of the Hybrid Autoregressive Transducer." In *Interspeech 2022, 23rd Annual Conference of the International Speech Communication Association, Incheon, Korea, 18-22 September 2022*, pp. 1646–1650. ISCA, 2022.

[Vaswani & Shazeer+ 17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin: "Attention is All you Need." In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5998–6008, 2017.

[Veselý & Ghoshal+ 13] K. Veselý, A. Ghoshal, L. Burget, D. Povey: "Sequence-discriminative training of deep neural networks." In *INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association, Lyon, France, August 25-29, 2013*, pp. 2345–2349. ISCA, 2013.

[Voigtlaender & Doetsch+ 15] P. Voigtlaender, P. Doetsch, S. Wiesler, R. Schlüter, H. Ney: "Sequence-discriminative training of recurrent neural networks." In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2015, South Brisbane, Queensland, Australia, April 19-24, 2015*, pp. 2100–2104. IEEE, 2015.

[Vyas & Madikeri+ 21] A. Vyas, S.R. Madikeri, H. Bourlard: "Comparing CTC and LFMMI for Out-of-Domain Adaptation of wav2vec 2.0 Acoustic Model." In *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia, 30 August - 3 September 2021*, pp. 2861–2865. ISCA, 2021.

[Wang & Sainath+ 21] P. Wang, T.N. Sainath, R.J. Weiss: "Multitask Training with Text Data for End-to-End Speech Recognition." In *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia, 30 August - 3 September 2021*, pp. 2566–2570. ISCA, 2021.

[Wang & Su+ 19] J. Wang, D. Su, J. Chen, S. Feng, D. Ma, N. Li, D. Yu: "Learning Discriminative Features in Sequence Training without Requiring Framewise Labelled Data." In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pp. 5696–5700. IEEE, 2019.

[Weng & Cui+ 18] C. Weng, J. Cui, G. Wang, J. Wang, C. Yu, D. Su, D. Yu: "Improving Attention Based Sequence-to-Sequence Models for End-to-End English Conversational Speech Recognition." In *Interspeech 2018, 19th Annual Conference of the International Speech Communication Association, Hyderabad, India, 2-6 September 2018*, pp. 761–765. ISCA, 2018.

[Weng & Yu+ 20] C. Weng, C. Yu, J. Cui, C. Zhang, D. Yu: "Minimum Bayes Risk Training of RNN-Transducer for End-to-End Speech Recognition." In *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, pp. 966–970. ISCA, 2020.

[Wiesler 17] S. Wiesler: "Optimization of discriminative models for speech and handwriting recognition." Ph.D. thesis, RWTH Aachen University, Germany, 2017.

[Wiesler & Richard+ 14] S. Wiesler, A. Richard, P. Golik, R. Schlüter, H. Ney: "RASR/NN: The RWTH neural network toolkit for speech recognition." In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014*, pp. 3281–3285. IEEE, 2014.

[Wynands 21] N.P. Wynands: "Global Normalization for Statistical Machine Translation." B.Sc. thesis, RWTH Aachen University, Germany, 2021.

[Wynands & Michel+ 22] N.P. Wynands, W. Michel, J. Rosendahl, R. Schlüter, H. Ney: "Efficient Sequence Training of Attention Models Using Approximative Recombination." In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022*, pp. 8002–8006. IEEE, 2022.

[Xiong & Droppo+ 17] W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, G. Zweig: "The microsoft 2016 conversational speech recognition system." In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*, pp. 5255–5259. IEEE, 2017.

[Yang & Zhou+ 23] Z. Yang, W. Zhou, R. Schlüter, H. Ney: "Investigating The Effect of Language Models in Sequence Discriminative Training For Neural Transducers." In *IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2023, Taipei, Taiwan, December 16-20, 2023*, pp. 1–8. IEEE, 2023.

[Yang & Zhou+ 24] Z. Yang, W. Zhou, R. Schlüter, H. Ney: "On the Relation Between Internal Language Model and Sequence Discriminative Training for Neural Transducers." In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2024, Seoul, Republic of Korea, April 14-19, 2024*, pp. 12627–12631. IEEE, 2024.

[Zeineldeen & Glushko+ 21] M. Zeineldeen, A. Glushko, W. Michel, A. Zeyer, R. Schlüter, H. Ney: "Investigating Methods to Improve Language Model Integration for Attention-Based Encoder-Decoder ASR Models." In *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia, 30 August - 3 September 2021*, pp. 2856–2860. ISCA, 2021.

[Zeineldeen & Zeyer+ 24] M. Zeineldeen, A. Zeyer, R. Schlüter, H. Ney: "Chunked Attention-Based Encoder-Decoder Model for Streaming Speech Recognition." In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2024, Seoul, Republic of Korea, April 14-19, 2024*, pp. 11331–11335. IEEE, 2024.

[Zeyer & Alkhouli+ 18] A. Zeyer, T. Alkhouli, H. Ney: "RETURNN as a Generic Flexible Neural Toolkit with Application to Translation and Speech Recognition." In *Proceedings of ACL 2018, Melbourne, Australia, July 15-20, 2018, System Demonstrations*, pp. 128–133. Association for Computational Linguistics, 2018.

[Zeyer & Bahar+ 19] A. Zeyer, P. Bahar, K. Irie, R. Schlüter, H. Ney: "A Comparison of Transformer and LSTM Encoder Decoder Models for ASR." In *IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2019, Singapore, December 14-18, 2019*, pp. 8–15. IEEE, 2019.

[Zheng & An+ 22] H. Zheng, K. An, Z. Ou, C. Huang, K. Ding, G. Wan: "An Empirical Study of Language Model Integration for Transducer based Speech Recognition." In *Interspeech 2022, 23rd Annual Conference of the International Speech Communication Association, Incheon, Korea, 18-22 September 2022*, pp. 3904–3908. ISCA, 2022.

[Zhou & Zheng+ 22] W. Zhou, Z. Zheng, R. Schlüter, H. Ney: "On Language Model Integration for RNN Transducer Based Speech Recognition." In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022*, pp. 8407–8411. IEEE, 2022.

[Zhu & Huang 20] X. Zhu, H. Huang: "End-to-End Amdo-Tibetan Speech Recognition Based on Knowledge Transfer." *IEEE Access*, Vol. 8, pp. 170991–171000, 2020.