School of Computing

Staffordshire UNIVERSITY

Prototyping a Language Model for the Transcription of German Medical Reports: An Empirical Study

Dissertation

For the Award of

Masters by Research (M.Res.) in Computing Science

Student: Patrick L. Lange

Research Area: Speech Recognition

Supervisors: Prof. Bernadette Sharp
Prof. David Suendermann-Oeft

In collaboration with DHBW Stuttgart

……..

2013 - 2015

Acknowledgements

I would like to thank so many people who have indirectly inspired me to conduct this research and produce this dissertation.

Firstly, I would like to express my sincere gratitude to my supervisors, Prof. Bernadette Sharp and Prof. David Suendermann-Oeft for their invaluable suggestions, assistance and guidance.

I would like to thank the DHBW Stuttgart for providing me with the necessary infrastructure and for partially funding my research project.

Furthermore, I would like to thank the Kaldi community for providing invaluable assistance with their toolkit used in my experiments.

Finally, I would like to thank Linguwerk GmbH for generously providing me with the required data as well as partially funding my research.

Abstract

The objective was to determine whether advanced language modelling techniques, particularly class-based N-gram language models and recurrent neural network based language models, can be effectively used to prototype a language model for German medical reports from radiology with small training data size.

For this purpose, a corpus with training data sets of size varying between 1k and 795k sentences was constructed. Then, word-based N-gram language models, the above mentioned advanced language models and language models created from combining several of these individual techniques with linear interpolation were trained on the different training data sets. Afterwards, these language models were used to rescore 1000-best hypotheses lists created with Kaldi using the state-of-the-art word based language model.

It was found that the combination of all three techniques, recurrent neural network language models, class-based and word-based N-gram language model, can achieve a relative word error rate improvement ranging between 8.23% and 15.30% for training data sets greater than 1k. Furthermore, it was found that the improvement achieved by using the combined language model is equal to the improvement achieved by doubling the training data size for training data sets of size 10k and greater. The absolute word error rates range from 28.91% to 7.31%.

The results demonstrate that advanced language modelling techniques in combination can be effectively used to prototype a language model for German medical reports from radiology using only small amounts of training data. However, at least 10k sentences or 500 reports should be used before applying advanced language modelling techniques is equally beneficial as doubling the amounts of training data.

Keywords: Language Modelling, Automatic Medical Transcription, Small Training Corpus

Table of Contents

List of Figures

List of Tables

Chapter 1

Statistical Language Modelling for Speech Recognition

1.1 What is Automatic Speech Recognition?

The task in Automatic Speech Recognition (ASR) is to transcribe a speech signal into the sequence that has been spoken. The complexity of this transcription task varies based on the particular application. Thus, Young (2008) categorises the speech recognition task into *command and control*, *dictation*, *transcription of recorded speech* and *interactive spoken dialogues* tasks. Besides the task itself, the supported vocabulary size can be used to categories the speech recognition task as small (up to 1k types), medium (up to 10k types), large (up to 100k types) and very large (more than 100k types) (Whittaker and Woodland 2001). In this research project focuses on the dictation task mainly using medium sized vocabularies.

1.2 What is Statistical Language Modelling?

In the late 1970s, Baker (1975) and Jelinek (1976) introduced a statistical approach to ASR. It has been implemented successfully in the speech recogniser *DRAGON System* (Baker 1975) and others since then. According

to Jelinek (2009), the *DRAGON System* has outperformed its competitors which were based on older approaches in a project whose goal was to find well performing speech recognisers. The project was started by the Advanced Research Projects Agency (ARPA) in 1971. In contrast to previous approaches, this approach models language with the help of statistics rather than grammars which had been previously used to syntactically and semantically describe language. As pointed out by Pereira (2000), linguists criticised the statistical approach but its practical success in ASR made it the current state-of-the-art.

As outlined in the previous section, the general task in speech recognition is to find the word sequence spoken in a given utterance. In the statistical approach, Jelinek (1976) formulates this problem as finding the *most likely* spoken sentence $\hat{w}$ given an acoustic signal A as described in Equation 1.1.

$$\hat{w} = \arg\max_{w} \Pr(w|A) \tag{1.1}$$

However, it is practically impossible to train a model which produces the most likely word sequence for every utterance because one cannot possibly collect enough data to cover every possible audio signal. Therefore, Jelinek (1976) uses Bayes' theorem (Equation 1.2) to transform Equation 1.1 as shown in Equation 1.3 and Equation 1.4. In Equation 1.5, $\Pr(A)$ can be dropped because it is constant to a changing w.

$$\Pr(X|Y) = \frac{\Pr(Y|X)\Pr(X)}{\Pr(Y)} \tag{1.2}$$

$$\hat{w} = \arg\max_{w} \Pr(w|A) \tag{1.3}$$

$$\overset{\text{Bayes'theorem}}{=} \arg\max_{w} \frac{\Pr(A|w)\Pr(w)}{\Pr(A)} \tag{1.4}$$

$$= \arg\max_{w} \Pr(A|w)\Pr(w) \tag{1.5}$$

Equation 1.5 splits the Equation 1.1 to two parts: $\Pr(A|w)$ modelling the probability of an audio signal for each possible sentence and $\Pr(w)$ representing the a priori likelihood of each possible sentence. The first part is

commonly called the *acoustic model*. The second part is called the *language model*. Both can be trained from training data.

A statistical language model commonly models the likelihood of the next word in a sequence (Bahl, Jelinek and Mercer 1983). Let w_1^n be an arbitrary word sequence of length n and w_i the i-th word in the sequence. Given the current sequence w_1^{n-1} also called context or history, the likelihood of the next word w_n is $\Pr(w_n|w_1^{n-1})$. Thus, the likelihood of a complete sequence is modelled as described in Equation 1.6.

$$\begin{aligned}\Pr(w_1^n) &= \Pr(w_1)\Pr(w_2|w_1)\Pr(w_3|w_1^2)\cdots\Pr(w_n|w_1^{n-1}) \\ &= \prod_{i=1}^{n}\Pr(w_i|w_1^{i-1})\end{aligned} \tag{1.6}$$

1.3 Language Modelling and Speech Recognition Software

There are several language modelling and speech recognition programs, tools and packages available for personal and organisational usages. Some popular language modelling and speech recognition toolkits are listed here.

Language Modelling:

- SRI Language Modeling Toolkit (SRILM)
- MIT Language Modeling toolkit (MITLM)
- Kyoto Language Modeling Toolkit (Kylm)
- RNNLM

Speech Recognition:

- Hidden Markov Model Toolkit (HTK)
- Kaldi
- pocketSphinx

- RWTH Aachen Automatic Speech Recognition System (RASR)
- Sphinx-4
- SRI International's Decipher

1.4 Motivation and Aim of the Study

An important practical problem of speech recognition is the training of well performing language models. Most language models widely used in ASR systems relies on large amounts of training data to reliably estimate the a priori probability of a given sentence. However, large quantities of suitable training data might not widely available in every domain. For example, the medical reports are mostly not freely available due to privacy policies and procedures.

In most real word applications, it is possible to record and store processed data for training better models in the future. However, this approach requires a reasonably well performing application to begin with. This research project focuses on rapid prototyping a language model from limited amounts of training data for the transcription of German medical reports particularly from radiology. Such a prototyped language model can then be used in the application to collect more data to train better models.

Thus, the research aim is to conduct an empirical analysis of language modelling techniques applied to limited amounts of German medical reports from radiology. The evaluation focuses on recurrent neural network (RNN) based language models, word and class-based 3-gram language models as well combinations of the three different techniques due to the lack of available language modelling toolkits supporting a larger variety of modelling approaches. The performance metric of interest is speech recognition accuracy. Thus, the research question asked in this study is formulated as

> "Can advanced language modelling techniques particularly RNN and class-based language models and their combinations in-

crease the speech recognition accuracy when trained on only limited amounts of training data consisting of German medical reports particularly from radiology when compared to the standard word-based language model?" .

1.5 Structure of the Document

This chapter gives a brief introduction of ASR and statistical language modelling, existing software and the aim of this study. Chapter 2 provides the literature review of the current state of statistical language modelling, flaws in the commonly used N-gram language model as well as proposed solutions to these issues. Chapter 3 describes the used research methodology and gives a brief overview of the data necessary to carry out this research project. Chapter 4 provides the background information to the performance metrics and language modelling techniques used in this study. In chapter 5, data collection decisions, data generation, the conducted experiments as well as the methods for analysing the results are presented. Chapter 6 and 7 present and discuss the produced results. The final chapter contains the conclusions which summarise the project work and findings. It also suggests some future work that can be useful to extend this study.

Chapter 2

Literature Review

2.1 Flaws of N-gram Models

N-gram language models cannot exactly compute the a priori probability of an arbitrary word sequence. They use only a very limited amount of of context to approximate the true a priori probabilities as explained in section 4.2 in more detail. Jelinek (1991) shows that humans can outperform N-gram language models when faced with the same task but also acknowledges that N-gram models perform reasonably well to be commonly used in ASR systems. In this section, the main flaws of N-gram language models are reviewed.

2.1.1 Choosing Appropriate Training Data

Research on language model adaptation techniques has shown that the performance of a language model depends on the topic of the discourse called domain (Kobayashi et al. 1998; Oku et al. 2013; Schlippe et al. 2013). Their finding relevant to this research is that the better the language model models the a priori probabilities of the task domain the better it performs. When the domain is large, with possible subtopics but training data is not sparse, such as in broadcast new, it can be beneficial to use language model adaptation techniques (Kobayashi et al. 1998; Oku et al. 2013; Schlippe et al. 2013). In this research, we focus on medical transcriptions particularly from radiology. This domain contains many domain-specific technical terms which do not occur in most textual data. Thus, a language model

representing the distribution of this domain should be trained on data from this domain, called in-domain data.

2.1.2 Expanding the Vocabulary

However, the information a language model can provide is also limited by the training data. Imagine a user wants to add his name or address to be recognised. An initial idea could be to add the new words to the dictionary which contains the word and its pronunciation and is used as a mapping between the acoustic model and the language model. Although the word could now be formed from the phoneme sequences given by the acoustic model, the language model contains no information about it because it has not been seen during training. It would be necessary to add the probability of the word and the probability of all N-grams it can occur in to the language model. This could be possible for synonyms where one can approximate the probability of the new word by one contained in the training data but not for an arbitrary word.

2.1.3 Limited Use of Context

As mentioned above, N-gram models limit the used context to the most recent $N-1$ words to model the likelihood of the next word. N-gram models are most commonly used with $N = 2$ (bigram model) or $N = 3$ (trigram model). Lau, R. Rosenfeld and Roukos (1993) point out that N-gram models cannot adapt to the style of a document due to their limited use of context and fail to model relationships seen in larger context. For example, one could argue that once the words *Golden Gate Bridge* have occurred in a conversation the probability of the word sequence *San Francisco* is higher. However, N-gram models can only model relationships of N consecutive words. Therefore, even in the sentence *"The Golden Gate Bridge is in San Francisco."* where the two sequences are close together, there is no trigram containing even only a part of both of them. Although this could be easily resolved by increasing the value of N, this would lead to another problem,

as described in the following.

2.1.4 Data Sparsity

The higher the value of N, the higher the amount of possible N-grams that could be created out of the training vocabulary. Let $|V|$ be the size of the vocabulary and N the N-gram order, then there are

$$\underbrace{|V| * |V| * \cdots * |V|}_{N \ times} = |V|^N \tag{2.1}$$

different N-grams that could be formed which is problematic for larger N.

Although most of these N-grams are grammatically incorrect or semantically useless and are therefore unlikely to occur, research on the Wall Street Journal Corpus (Ronald Rosenfeld 2005) has shown that 21% of the trigrams in the test data have been unseen in 38M tokens of training data when the vocabulary is limited to the most frequent 5k types and 32% when limited to the most frequent 20k types. In addition to these unseen N-grams, there are N-grams which occur only very rarely in the training data and therefore cannot provide a reliable estimation of their probability. In Natural Language Processing (NLP) this problem of not being able to model language due to unseen or underrepresented events in the training data is called data sparsity, data sparseness or data paucity (Ben Allison, David Guthrie and Louise Guthrie 2006). D. Guthrie, B. Allison et al. (2006) and D. Guthrie, L. Guthrie and Wilks (2009) identified data sparsity as the main problem for further advances in NLP and pinpoint the origin of data sparsity in the assumption "[...] that language is a system of rare events, so varied and complex, that we can never model all possibilities." (D. Guthrie, B. Allison et al. 2006).

2.2 Solutions

So what can be done to solve the above mentioned flaws in standard N-gram models? Across various research studies in the field, two major approaches

can be identified. One the one hand, researchers suggest to gather more data. On the other hand, more advanced language modelling techniques have been developed.

2.2.1 Increase the Corpus Size

Research on N-gram coverage with increasing corpus size has shown that even with a 1.48B token corpus the 3-grams seen during training do not cover 100% of 3-grams in the test data (Ben Allison, David Guthrie and Louise Guthrie 2006). However, results obtained by Ben Allison, David Guthrie and Louise Guthrie (2006) also show that the coverage for 3-grams increases almost linearly when the corpus size is increased from 160k tokens (6% coverage) to 1.46B tokens (72% coverage). However, it seems unlikely that the increase stays linear for larger training corpora because that would mean that more than 100% coverage is achievable. For 2-grams the coverage starts at 33% at a training corpus size of 160k words but the increase in coverage slows down after 26M tokens (84% coverage) to cover 95% of the 2-grams in the test data with a training corpus size of 1.48B. A similar trend can be observed for 1-grams (160k : 86%, 1M : 96% , 1.48B : 99%). These findings suggest that the coverage for 3-grams can be increased with more data but that no full coverage can be achieved.

Combined with the findings of Ronald Rosenfeld (1995) that better N-gram coverage correlates with better speech recognition accuracy, the approach to collect more data is very promising to improve language modelling and solving the data sparsity problem.

However, another key finding of Ronald Rosenfeld (1995) is that increasing the size of the vocabulary past a certain point decreases the speech recognition accuracy due to added acoustic confusability. The given explanation is that while a bigger vocabulary allows for more N-grams and thus a bigger N-gram coverage, a larger vocabulary also means that the acoustic component of the speech recogniser has a more difficult task to distinguish between the individual words in the vocabulary and therefore introduces additional

errors.

Furthermore, training large scale language models on billions of tokens poses computational problems during training as well as during decoding time. This is due to the size of the training data which has to be processed and the size of the resulting language model. Brants, Popat et al. (2007) implemented a distributed approach using the *MapReduce* programming model (Dean and Ghemawat 2008). In addition, they propose a simple and easily computable smoothing technique to deal with unseen N-grams which further reduces the training time. With the suggested implementation, Brants, Popat et al. (2007) managed to train a language model on 1.8T tokens in 1 day using 1500 machines in parallel. In comparison, training a model on 237M tokens took 20min using 100 machines with the same approach. The sizes of the resulting language models are 2GB for the 237M tokens model and 1.8TB for the 1.8T tokens model. Brants, Popat et al. (2007) had to also modify the decoder architecture to use a distributed approach to use such large models directly during decoding.

Besides the issue of training and using such large models, the problem of where to get all the required training data from remains. Kilgarriff and Grefenstette (2003) were one of the first to suggest that the World Wide Web can be used as a corpus. Since then, different techniques to efficiently crawl the web have been developed (Baroni and Ueyama 2006; de Groc 2011; Suchomel and Pomikálek 2012). Furthermore, many text corpora have been built from the web such as the *English Gigaword* (Graff and Cieri 2003), Google's *Web 1T 5-gram* (Brants and Franz 2006) or the *WaCky corpora* (Baroni, Bernardini et al. 2009). However, as de Groc (2011) notes, while crawling the web for text corpora it is important to evaluate the quality of the web. Research studies on the quality of corpora by crawling the web identified that not only the used crawling technique but also post-processing and quality control are essential to create a high quality corpus (Biemann et al. 2013; Schäfer, Barbaresi and Bildhauer 2013; Versley and Panchenko 2012).

Although the web provides large amounts of textual data, it is important to choose training data closely matching the target domain as outlined in subsection 2.1.1. Various research studies have shown that adaptation techniques are useful in the case where a domain contains several heterogenous subdomains such as newspaper articles (Lau, R. Rosenfeld and Roukos 1993; Schlippe et al. 2013). Research suggests that intelligent selection of training data does also work for homogenous domains without several distinct subdomains but requires a substantial amount of in-domain data for making the selection (Moore and Lewis 2010).

However, all of the above mentioned techniques are based on the premise that data relevant to the task domain is available. This might be true for popular domains such as newswire text or blog posts but what if the web does not provide enough data?

2.2.2 Train Advanced Language Models

N-grams are not perfect at modelling language as has been pointed out in section 2.1. A study by Jelinek (1991) shows that a human can outperform N-gram based language models when challenged with the task to predict the next word in a sequence. However, the study also shows that N-gram language models perform reasonably well and are not that easily outperformed. Despite that, advanced language modelling techniques have been developed which commonly address one of the flaws outlined in section 2.1. They can be categorised as either an extension to the standard word-based N-gram language model or as completely new technique.

The class-based N-gram model approach (Brown et al. 1992) belongs to the first category. It suggests to group words into classes based on the frequency of their co-occurrence with other words and train a N-gram model over the classes instead of the words. With this approach, one hopes to reduce the data sparsity problem because less N-grams can be generated from a smaller class-based vocabulary (Equation 2.1). However, they could only show a reduced size of the resulting language model but not better performance.

Additional research improved on the automatic clustering technique (Bahl, Jelinek and Mercer 1983) and confirmed that class-based N-gram models do not perform better than N-gram models (S. Martin, Liermann and Ney 1998). However, a key finding of S. Martin, Liermann and Ney (1998) was that class-based N-gram models when interpolated with word-based N-gram models can outperform the standard word-based N-gram language model Furthermore, they found that the clustering technique optimised for a bigram class model outperforms the clustering technique optimised for a trigram class model when used to train a trigram class-based model from a a small corpus of 1M tokens. Ward and Issar (1996) point out that the above mentioned class-based N-gram models have limitations when faced with linguistically motivated classes such as *book titles* or *times*. They argue that classes should not contain single words but but should be able to model word sequences. Their proposed solution to let classes model finite-state networks instead of single words is linguistically well motivated but does not answer the question no how to come up with such classes automatically. In their experiment, they utilised database of a Library Information System to generate the classes but the approach would not be possible if the finite-state networks are not given by an external source.

Another alternation of the N-gram language model is the distance-k skip N-gram language model proposed by D. Guthrie, B. Allison et al. (2006) and Siu and Ostendorf (2000). The propose approach allows for a total of k words to be skipped in the context while still producing N-grams to combat the data sparsity problem. This allows for more N-grams to be generated from the same word sequence. Although it might seem that most of these N-grams might not be particularly useful, D. Guthrie, B. Allison et al. (2006) show that distance-k skip N-gram models can have a similar positive effect on N-gram coverage like increasing the size of the training corpus. When applied to Gigaword corpus (Graff and Cieri 2003) distance-4 skip 3-grams trained on 50M words achieve the same coverage on the test data as a standard 3-gram model trained on 4 times more data (D. Guthrie, B. Allison et al. 2006). However, this experiment was carried out on the

domain of newswire text which as mentioned above has several subdomains. Moreover, they only report N-gram coverage but not the performance in a speech recognition task.

A similar approach is a distance-d bigram model (Brun, Langlois and Smaïli 2007; Huang et al. 1993; Simons, Ney and S. C. Martin 1997) which incorporates long distance information to solve the flaw of limited use of context in N-gram language models. The proposed approach models the probability of the next word in a sequence based on the word d positions before it. Research shows that a distance-d bigram model when interpolated with a standard N-gram model can slightly increase the recognition accuracy (Brun, Langlois and Smaïli 2007; Simons, Ney and S. C. Martin 1997). However, Huang et al. (1993) show that the performance decreases when distance d is increased. However, this research also indicates that the history does contain valuable information but that it needs to be selected carefully. Zhou and Lua (1998) proposes such a selection based on the correlation of the word pairs so that well associated pairs remain in the model. The premise for this approach is that as soon as the first word in the pair occurs in the context, the probability of the second word increases. Such a word pair is called a trigger pair. This approach, called distance-dependent trigger model, shows also an improvement in performance when interpolated with a standard N-gram model. The approach is similar to distance-independent trigger models introduced by Lau, R. Rosenfeld and Roukos (1993) and Zhou and Lua (1998) which according to Su, Jelinek and Khudanpur (2007) have gained more popularity since better training algorithms for the proposed architecture (Wu and Khudanpur 2000) have been developed.

The cache-component suggested by Kuhn and Mori (1990) which is complementarily used with a N-gram language model models only the special case of trigger pair in which both words are the same. Essentially, the premise is that when a word has been used in the recent past it is more likely to be used again than the overall frequency in the language or a N-gram model would suggest. Kuhn and Mori (1990) show that these so called self-triggers

can drastically increase the language model performance on a corpus containing different domains but also question the efficiency of the approach when trained on a homogenous domain.

A completely different language modelling technique is the random forest language model (Xu and Jelinek 2004). It uses a decision tree to randomly cluster the history to provide a better use of the available context. This decision trees do not perform well on unseen data when applied individually. Thus, a combination of many decision trees, called a forest, is used. The approach allows for a larger use of history than N-gram language models. Research has shown that such random forest language models can reduce the number of unseen events in test data when compared to the standard N-gram language model (Xu and Jelinek 2004). Xu and Jelinek (2004) report a similar increase in performance as above mentioned techniques but without interpolating with a N-gram model. However, Su, Jelinek and Khudanpur (2007) points out that random forest language models have practical problems due to space complexity when trained on larger corpora.

Another language model architecture using feedforward neural networks (FFNNs) has been introduced by Bengio et al. (2003), Schwenk (2007) and Schwenk and Gauvain (2003). FFNNs try to solve the data sparsity problem by representing words in a continuous space instead of the discrete space N-gram models use. The benefit of this approach is that an unseen sequence of words gets assigned a higher probability if it is made out of words that are closer to other words in the continuous space which form an already seen word sequence (Bengio et al. 2003). Therefore, they can take advantage of a longer context without being confronted with the data sparsity problem as fast as the N-gram models. It has been shown that FFNN based language models can slightly outperform standard N-gram language models (Schwenk 2007) and that they are well suited for small amounts of training data (Schwenk and Gauvain 2003). Although the approach allows for a larger context to be used than with N-gram models, the amount of history used is still limited to a fixed size.

Mikolov (2012) proposes a RNN based language model which removes the constraint of a fixed sized context. RNN based language models can use an unlimited amount of history by carrying the hidden layer of the network forward to the current time step. Furthermore, Mikolov (2012) suggests computational optimisations such reducing the dimensionality of the input to the network by clustering words into classes to speed up the training process. The study shows that RNN based language models outperform the standard N-gram language models as well as the FFNN based language model even when using such optimisations.

Most of the above mentioned techniques are evaluated individually and are compared against a N-gram model baseline. However, some of the models try to solve the same flaw in N-gram models by providing very similar kind of new information such as longer context. This has also been pointed out by J. T. Goodman (2001) who suggest to jointly study language modelling techniques. This proposal is backed up by the fact that a fair amount of the techniques discussed above can only show improvement over standard N-grams when interpolated with them. In addition, J. T. Goodman (2001) shows that a combination of more than two language modelling techniques improves the performance even more. However, the study did not include the only recently introduced techniques such as RNN based language models (Mikolov 2012). Another important finding of J. T. Goodman (2001) is that the improvement provided by some techniques tend to decrease as the amounts of training data increases. This finding suggest that it is important to take the size of the training corpus into account when evaluating language modelling techniques jointly.

2.3 Varying Training Corpus Size

Research on the performance scaling of acoustic models with increasing amounts of audio data has shown that there seems to be a linear relationship between the performance and the logarithm of the training data size in hours. It would be worthwhile to investigate whether there is a similar

relationship between training data size and the performance of combinations of different language modelling techniques. Although considerable research has been devoted to develop advanced language models and to study N-gram coverage for different training data size, rather less attention has been paid to how combinations of these techniques scale with training data.

2.4 Summary

Nowadays, statistic language models are a key component of ASR systems. The widely used N-gram language models only provide a suboptimal estimation of the necessary a priori probability of an arbitrary sentence and suffer mainly from limited use of history and data sparsity. Research has shown to follow two different ways of improving the performance of the language model component either by increasing the amounts of training data or by developing more advanced models.

The purpose of this research project is to answer how combinations of various language modelling techniques perform with varying amounts of training data size. The research focuses on techniques which require no manual supervision as well as on very small training corpus sizes to simulate domains in which data is not widely available and gathering more data is no immediate option.

Chapter 3

Research Methodology

The research methodology is important for carrying out a successful study. Choosing an appropriate approach, methods of data collection and methods of data analysis is essential for creating reliable and valid results. This decisions depend on the type of research question(s) to be answered by the study but are also influenced by the research area and the chosen philosophical approach.

The approach can either be deductive or inductive (Hyde 2000). According to Hyde (2000), the starting point in a deductive approach is a general theory from which hypotheses are deduced. These hypotheses are then tested using the gathered data. Finally, conclusions about the truth of the initial theory can be drawn from the outcome of the hypotheses tests. In contrast to a deductive approach which focuses on theory testing, an inductive approach aims at generating new theories. The starting points in an inductive approach are the observations. They are analysed for specific patterns which are generalised into hypotheses. These hypotheses are then used to formulate a new theory.

This research uses a deductive approach because the theory of statistical language modelling is well established. The research question of this study outlined in section 1.4 is motivated by the findings of existing researches as described in chapter 1 and chapter 2. Therefore, a deductive approach is the logical choice because the starting point is a theory which is to be proven by hypotheses testing.

The methods for data collection and data analysis can be qualitative, quantitative or in certain cases a mix of both (Creswell 2013, pp. 11–12). Although qualitative research is commonly associated with an inductive approach and quantitative research with a deductive approach, there are no set rules that forbid other combinations.

Quantitative research methods are based around proving hypotheses using mathematical and statistical means (Creswell 2013). Quantitative research is usually interested in whether two variables correlate. The variable suspected to be the cause, called the independent variable, is modified by the researcher and the effect, called dependent variable, is observed. For example, a typical hypothesis in a quantitative research could be "The older a human being is, the bigger it is.". In this example, the age is the independent variable and the height is the dependent variable. In a quantitative research, it is important to choose samples randomly, make use of a control group whenever possible, design the research in a repeatable way and change only one independent variable at a time to produce valid and reliable results.

One advantages of quantitative researches methods is that they filter out external factors and therefore produce unbiased results. Furthermore, the answers after the statistically analysis are either yes or no. However, clear answers do not exist in every research area. Thus, quantitative research methods are not practical in those fields. Finally, it is important to keep in mind that correlation between two variables does not imply causation.

Qualitative researches seek to explore phenomena. In contrast to quantitative researches, they try to describe and explain relationships and experiences instead of quantifying them exactly (Kaplan and Maxwell 2005, pp. 30–31). The research questions in qualitative research are commonly more open-ended and cannot be answered by a clear yes or no.

Advantages of qualitative research methods are that they can give ideas about relationships, individual experiences or group norms. This is especially useful in complex research areas such as studying human behaviour.

Furthermore, qualitative research can be more flexible because the methods do not need to be as highly structured as quantitative methods to produce valid and reliable data. However, qualitative data cannot be analysed in the same way as quantitative data. Therefore, the answers qualitative researches give are only trends. Finally, qualitative researches are often not exactly repeatable due to the flexible nature of the research methods.

This work uses quantitative research methods because one of its objectives is to quantify the effect of different language modelling techniques on the chosen performance metric. The performance metric will be the dependent variable in the analysis. The independent variables will be varied and tested one by one. The type of applied research design is experimental. This means that the data are generated in clearly structured experiments (Creswell 2013, p. 13). The analysis of the results is focused on formulating and testing null hypotheses. The validity and reliability of the analysis is ensured by using statistical significance testing and clear structuring of the experiments.

In the following, a brief overview of the data which needs to be collected, the data of the experiments which are analysed and the method of analysing this output data is given. A more detailed explanation of the data collection, experiments and analysis of results can be found in chapter 5.

Data collection

Dictionary

- Used to train the acoustic model which is used in the speech recogniser
- Must be in the same language (German) as in-domain data
- Used to train grapheme to phoneme model which is used to produce dictionary for the in-domain data

Transcribed audio data

- Used to train the acoustic model which is used in the speech recogniser
- Must be in the same language (German) and of same audio conditions (office setting) as in-domain data

In-domain data

- Used to create training sets as well as a development and an evaluation set

Audio recordings of development and evaluation set

- Speech recognition is performed on this audio data to measure the performance of evaluated language modelling techniques
- Audio conditions must match the audio condition of the data used to train the acoustic model

Experimental data to analyse

N-best hypotheses of development set

- Used to tune the weights between acoustic model and language model in the speech recogniser
- Used to evaluate best possible results achievable by rescoring

Rescored n-best hypotheses of development set

- Used to optimise the weights for the combination of language modelling techniques

Rescored n-best hypotheses of evaluation set

- Used to verify that the results of the development set did not happen due to overfitting

Data analysis

Hypotheses

- Hypotheses on the predicted outcomes are formulated as null-hypotheses for each experiment
- Are formulated before the experiments are run
- The level of statistical significance to reject the null-hypotheses is chosen

Statistical significance testing

- Appropriate tests of statistical significance are used to test the formulated hypotheses

Chapter 4

Statistical Language Modelling

4.1 Evaluation

Besides training and using a language model, it is very important to be able to measure its performance. The only way to improve on existing language models is to compare new models against existing once regarding a metric. This evaluation of the performance of different language models is commonly done by using either perplexity or word error rate (WER) as metric. Both metrics have advantages as well as drawbacks which will be concisely covered in the following sections.

4.1.1 Perplexity

Jelinek et al. (1977) introduced perplexity as a measure of speech recognition difficulty. Let $(w_n)_{n\in\mathbb{N}}$ be the n^{th} word in a sequence, $T = \{w_1 w_2 \dots w_L\}$ a test of length L and M a language model providing the probability $Pr_M(w_i|C_i)$ of the next word w_i given a context C_i. Perplexity is then defined as described in Equation 4.1.

$$PP_T(M) = \left(\prod_{i=1}^{L} Pr_M(w_i|C_i) \right)^{-\frac{1}{L}} \tag{4.1}$$

Perplexity is related to the cross entropy $H(T, M)$ between the test set T and the language model M. Since the distribution which created the test set is unknown, only a *Monte Carlo estimate* of the true cross entropy can be computed (Equation 4.2). When the transpositions outlined

in Equation 4.3 are applied to Equation 4.1 this relation becomes clearer. If the cross entropy between some a test set T and a language model M is $H(T, M) = 7$, this means the language model encodes every word in the test set with on average 7 bits, then the perplexity on the same test set and language model is $PP_T(M) = 2^7 = 128$.

$$H(T, M) = -\frac{1}{L} \sum_{i=1}^{L} \log_2 Pr_M(w_i|C_i) \tag{4.2}$$

$$\begin{aligned} PP_T(M) &= \left(\prod_{i=1}^{L} Pr_M(w_i|C_i) \right)^{-\frac{1}{L}} \\ &= \sqrt[L]{\frac{1}{\prod_{i=1}^{L} Pr_M(w_i|C_i)}} \\ &= 2^{-\frac{1}{L} \sum_{i=1}^{L} \log_2 Pr_M(w_i|C_i)} \end{aligned} \tag{4.3}$$

As mentioned above, perplexity has been introduced as metric for speech recognition difficulty. Equation 4.1 shows that perplexity depends on the used language model and can therefore be used to compare different models. The language model which results in the lowest value for speech recognition difficulty can be seen as the best model. Due to the relationship to cross entropy, this the model which results in the lowest perplexity value is the model which compresses the data best and is in some sense the model closet to the real model which generated the data. Although, Chen, Beeferman and Ronald Rosenfeld (1998) and J. T. Goodman (2001) show that perplexity mostly correlates good with speech recognition performance, Iyer, Ostendorf and Meteer (1997) and S. C. Martin, Liermann and Ney (1997) show that this is not always true. Chen, Beeferman and Ronald Rosenfeld (1998) add that the good correlation between perplexity and the performance of an ASR system is mostly true for N-gram models but sometimes fails for more advanced models for example models which use longer context information such as cache based models.

Researchers tend to report improvements in relative perplexity reduction. However, as shown in Table 4.1, the same relative perplexity reduction translates in different cross entropy reductions. Therefore, it is not correct to report on relative perplexity reduction. It would be more appropriate to report absolute values for perplexity or in case relative improvements shall be reported cross entropy should be used.

PP	PP after reduction	Relative PP reduction	Entropy [bits]	Entropy after reduction	Relative entropy reduction
2	1.4	30%	1	0.49	51%
20	14	30%	4.32	3.81	11.8%
100	70	30%	6.64	6.13	7.7%
200	140	30%	7.64	7.13	6.7%
500	350	30%	8.97	8.45	5.8%
2000	1400	30%	10.97	10.45	4.7%

Table 4.1: Constant 30% perplexity reduction translates to variable entropy reduction. Taken from Mikolov (2012, p.14).

Advantages of perplexity:

- Simple to evaluate (Does not require speech recognition system)
- Mostly good correlation between perplexity and ASR system performance (for N-gram models)

Drawbacks of perplexity:

- Not always good correlation between perplexity and ASR system performance (for more advanced models)
- Does not account for interaction with other ASR system components (Acoustic model, Dictionary)
- Relative perplexity improvement not comparable

In this research project, perplexity is used to quickly measure the performance of the trained language models and to find best interpolation weights

for combinations of language modelling techniques. However, we do not rely on perplexity to report the actual results of the speech recognition accuracy.

4.1.2 Word Error Rate

The WER is the metric used to measure the application performance of a speech recogniser. It compares the output of the speech recogniser, the hypothesis, with the word sequence which has been spoken, the reference.

Let d_L be the Levensthein distance (Levenshtein 1966)) (Equation 4.4) where S is number of substitutions, D deletions and I insertions applied to the hypothesis *hyp* to edit it into the reference *ref*.

$$d_L(\ hyp, ref\ ) = \min(S + D + I) \tag{4.4}$$

Let N_{ref} be the length of the reference. The WER(hyp, ref) between the hypothesis and the reference is then defined as

$$\text{WER}(\ hyp, ref\ ) = \frac{d_L(\ hyp, ref\ )}{N_{ref}}. \tag{4.5}$$

WER directly measures the quality of the speech recogniser as whole. It measures the performance of all of the used components (acoustic model, dictionary and language model) and allows to measure the interaction of different language models with the other components when only the language model is changed. However, the WER is strongly influence by frequently occurring words which are often function words and often to not contribute much information to speech understanding. Furthermore, it also counts the substitution with a word of similar meaning like a substitution with a word of different meanings. A modified version of the WER, NIST WER tolerates substitutions between words with the same meaning. The task of the speech recogniser is to transform the spoken speech to text and not understanding the meaning. Thus, the WER is the most common metric to measure the accuracy of a speech recogniser. However, it is important

to compare different language modelling techniques on the same task with the same ASR system configuration when using WER as metric.

Advantages of WER:

- Is the final metric to be optimised for ASR systems
- Easy to compute when hypotheses and matching references are available

Drawbacks of WER:

- WER results tend to be noisy e.g. audio conditions play a role
- Speech recognition system is needed
- Frequent, uninformative words are over-emphasised
- Substitution with word of similar meaning produces error

In this research, WER is used to report the performances of the different language modelling techniques because it reflects the metric used to optimise the ASR system.

4.2 N-gram Models

As mentioned in section 2.1, N-gram model limits the length of the context used to estimate the probability of the next word in a word sequence. They limit the context to the most recent $N-1$ words which is equivalent to the approximation in Equation 4.6 (Bahl, Jelinek and Mercer 1983).

$$\Pr(w_1^n) = \prod_{i=1}^{n} \Pr(w_i|w_1^{i-1}) \approx \prod_{i=1}^{n} \Pr(w_i|w_{i-N+1}^{i-1}) \tag{4.6}$$

The probabilities of the individual N-grams are trained using the counts of their occurrence. The likelihood of a word w_i in the context w_{i-N+1}^{i-1} is

computed as

$$Pr(w_i|w_{i-N+1}^{i-1}) = \frac{Pr(w_{i-N+1}^{i})}{Pr(w_{i-N+1}^{i-1})} = \frac{c(w_{i-N+1}^{i})}{c(w_{i-N+1}^{i-1})} \tag{4.7}$$

where $c(s)$ denotes the number of times the string s occurred.

Smoothing is used to assure to assign non-zero probabilities to N-grams not present in the training data. This is achieved by redistributing a certain amount of probability mass from seen to unseen events.

A simple smoothing technique is plus-one smoothing. It assumes that every N-gram occurs once more than it actually did in the training data (Jeffreys 1948; Lidstone 1920). This assures that no N-gram probability is zero but assigns the same probability to every unseen N-gram. Applying plus-one smoothing to Equation 4.7 would result in

$$Pr_{+1}(w_i|w_{i-N+1}^{i-1}) = \frac{c(w_{i-N+1}^{i}) + 1}{c(w_{i-N+1}^{i-1}) + |V|} \tag{4.8}$$

where $|V|$ is the number of words in the vocabulary.

There are many different smoothing techniques such as additive smoothing (Jeffreys 1948; Johnson 1932; Lidstone 1920), Jelinek-Mercer Smoothing (Jelinek 1980), Katz Smoothing (Katz 1987) or absolute discounting (Ney and Essen 1991; Ney, Essen and Kneser 1994). However, the smoothing techniques that have shown to produce the best results are Kneser-Ney smoothing (Kneser and Ney 1995) shown by J. T. Goodman (2001) and a modified version of Kneser-Ney smoothing proposed by and shown to outperform Kneser-Ney smoothing by Chen and J. Goodman (1999). Therefore, Kneser-Ney (KN) smoothing and modified Kneser-Ney (mKN) smoothing are used within this study.

Kneser and Ney (1995) observed that most smoothing techniques can be formulated as backing-off models of the form

$$Pr(w|h) = \begin{cases} \alpha(w|h) & \text{if } c(h\ w) > 0 \\ \gamma(h)\beta(w|\hat{h}) & \text{otherwise} \end{cases} \tag{4.9}$$

where h denotes the history, $\hat{h}$ denotes a less specific history. For example, for a N-gram model $\hat{h}$ would be the history of the N'-model with $N' = N - 1$. α is some reliable estimation of the probability of seen events and $\gamma(h)\beta(w|\hat{h})$ is the estimation for the remaining unseen events according to a less specific distribution β and a normalisation factor γ to ensure that $Pr(w|h)$ sums to 1.

Kneser-Ney smoothing for N-gram models is then defined as

$$Pr_{KN}(w_i|w_{i-N+1}^{i-1}) = \begin{cases} \frac{\max(c(w_{i-N+1}^{i})-D,0)}{c(w_{i-N+1}^{i-1})} & \text{if } c(w_{i-N+1}^{i}) > 0 \\ \gamma(w_{i-N+1}^{i-1}) Pr_{KN}(w_i|w_{i-N+2}^{i-1}) & \text{otherwise} \end{cases} \tag{4.10}$$

with $0 \leq D \leq 1$ and

$$Pr_{KN}(w_i|w_{i-N+2}^{i-1}) = \frac{C_{1+}(\cdot w_{i-N+2}^{i})}{\sum_{w_i} C_{1+}(\cdot w_{i-N+2}^{i})} \tag{4.11}$$

where $C_{1+}(\cdot w_{i-N+2}^{i})$ denotes the number of unique words preceding the word sequence w_{i-N+2}^{i}. In this approach, an absolute value D is subtracted from all seen N-grams and redistributed to the unseen N-grams. This is identical to absolute discounting (Ney and Essen 1991; Ney, Essen and Kneser 1994). The novelty of this approach is to actually change the less specific distribution β as shown in Equation 4.11 which would normally be $\beta(w|\hat{h}) = Pr(w|\hat{h})$ for a regular backing-off model. If $\hat{h} = w_{i-N+2}^{i-1}$, then this would be a regular backing-off N-gram model.

The modification Chen and J. Goodman (1999) proposed is to not only use a single absolute discount value D but to use three values D_1 D_2 and D_{3+} for N-grams that respectively occur once, twice or more than three times.

4.3 Class Based Models

Class based N-gram models use N-grams of word classes instead of N-grams of words to estimate the probability $Pr(w|h)$ of the next word w following

a given history h. For that reason, a class mapping

$$G : w \rightarrow G(w)$$

which maps each word into a class $G(w)$ is required.

Given such a class mapping G, several different class based N-gram models can be constructed to compute $Pr(w_i|w_{i-n+1}^{i-1})$ (Whittaker 2000), for example as follows:

$$Pr(w_i|G(w_i))Pr(G(w_i)|G(w_{i-N+1})\dots G(w_{i-1})) \tag{4.12}$$

$$Pr(w_i|G(w_i))Pr(G(w_i)|w_{i-N+1}^{i-1}) \tag{4.13}$$

$$Pr(w_i|G(w_{i-N+1})\dots G(w_{i-1})). \tag{4.14}$$

In this study, we focus on class based N-gram model described by Equation 4.12 introduced by Brown et al. (1992). The 1-gram component of this class based N-gram model are estimated as

$$Pr(w|G(w)) = \frac{c(w)}{c(G)} \tag{4.15}$$

where c(G) is the number of words in the training data for which the class is G. Let G_i be the class of the i^{th} word in a word sequence and

$$G_i^j = G_i G_{i+1} \dots G_j \; with \; i < j$$

The N-gram component $Pr(G_i|G_{i-N+1}^{i-1})$ is then estimated analog to the word based N-gram model as

$$Pr(G_i|G_{i-N+1}^{i-1}) = \frac{Pr(G_{i-N+1}^{i})}{Pr(G_{i-N+1}^{i-1})} = \frac{c(G_{i-N+1}^{i})}{c(G_{i-N+1}^{i-1})}. \tag{4.16}$$

Thus, $Pr(w_i|w_{i-n+1}^{i-1})$ can be estimated as

$$\begin{aligned} Pr(w_i|w_{i-n+1}^{i-1}) &= Pr(w_i|G(w_i))Pr(G_i|G_{i-N+1}^{i-1}) \\ &= \frac{c(w_i)}{c(G_i)} * \frac{c(G_{i-N+1}^{i})}{c(G_{i-N+1}^{i-1})}. \end{aligned} \tag{4.17}$$

In this research project, we use the $O(|V| * |G|^2)$ algorithm described by Brown et al. (1992) where $|V|$ is the number of words in the vocabulary and $|G|$ the number of desired classes.

Algorithm to produce the class mapping:

Initialisation:

- Order the words in the vocabulary by there frequency starting with the most frequent word
- Assign the first $|G|$ words each to an own class

Step 1:

- Assign the $(|G|+1)^{st}$ most probable word to a new class
- Merge the class pair of the resulting $|G|+1$ classes which results in the least loss of average mutual information

Step N:

- Assign the $(|G|+N)^{st}$ most probable word to a new class
- Merge the class pair of the resulting $|G|+1$ classes which results in the least loss of average mutual information

Termination:

- After $|V|-|G|$ steps, each word is assigned to one of the $|G|$ classes

For information on the computation of the average mutual information remaining after merging two classes see Brown et al. (1992).

4.4 Recurrent Neural Network Based Models

RNN based models introduced by Mikolov et al. (2010) use a completely different architecture than N-gram models. The architecture is shown in Figure 4.1. The input into the network consists of the vectors $w(t)$ and $s(t-1)$. $w(t)$ represents the current word w_t as $|V|$ dimensional vector with all components set to zero except the one representing the word w_t in the vocabulary V of size $|V|$ which is set to 1. $s(t-1)$ is the output

of the hidden layer from the previous time step. The output layer $y(t)$ is of the same size as $w(t)$ but y_i contains $Pr(w_{t+1}|w_t s(t-1)) : w_{t+1} = v_i$ where v_i denotes the word of the vocabulary V that corresponds to the i^{th} component in the encoding.

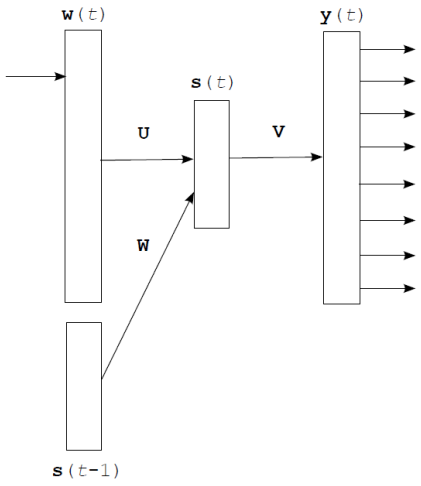


Figure 4.1: Simple recurrent neural network. Taken from Mikolov (2012, p.29)

U and W are weight matrices between the input and the hidden layer and V is a weight matrix between the hidden layer and the output layer. The output values of the hidden layer s and the output layer y are then computed as

$$s(t) = f(Uw(t) + Ws(t-1)) \tag{4.18}$$

$$y(t) = g(Vs(t)) \tag{4.19}$$

with

$$f(z) = \frac{1}{1+e^{-z}}, g(z_i) = \frac{e^{z_i}}{\sum_k e^{z_k}} \tag{4.20}$$

where $f(z)$ and $g(z)$ are sigmoid and softmax activation functions. The purpose of $g(z)$ is to ensure that all outputs are greater than zero and sum to 1 as expected from a valid probability distribution.

Backpropagation algorithm to train RNN:

Initialisation:

- Initialise weight matrices U, V and W to small random numbers
- Set time counter t = 0, initialize state of the neurons in the hidden layer s(t) to 1

Step N:

- Set time counter t = N
- Load the current word w_t in the input layer $w(t)$
- Load the output of the hidden layer of the previous step $s(t-1)$ to the input layer
- Compute $s(t)$ and $y(t)$ as described in Equation 4.18 and Equation 4.19
- Compute gradient of error e(t) in the output layer
- Propagate error back through the neural network and update U, V and W

Termination:

- The RNN based language model is trained after all training samples $t = 1 \dots t$ have been visited.

The function which is aimed to maximise is the likelihood of the training data and the gradient $e_o(t)$ of the error $e(t)$ in the output layer is computed as follows:

$$e_o(t) = d(t) - y(t) \tag{4.21}$$

where $d(t)$ is the word w_{t+1} that should have been predicted using the same encoding as $w(t)$.

Let α be the learning rate and β the regularisation parameter used to keep the weights small which is preferred (Mikolov 2012). Then V gets updated as described in Equation 4.22.

$$V(t+1) = V(t) + s(t)e_o(t)^T\alpha - V(t)\beta. \tag{4.22}$$

Then, the gradient e_o of the error on the output layer is propagated back to the hidden layer as

$$e_h(t) = d_h(e_o(t)^T V, t) \tag{4.23}$$

$$d_{hj}(x,t) = xs_j(t)(1 - s_j(t)) \tag{4.24}$$

where $d_h()$ is applied element-wise. The above propagated gradient e_h of the error on the hidden layer is then used to update U and W as follows:

$$U(t+1) = U(t) + w(t)e_h(t)^T\alpha - U(t)\beta \tag{4.25}$$

$$W(t+1) = W(t) + s(t-1)e_h(t)^T\alpha - W(t)\beta \tag{4.26}$$

The backpropagation algorithm described above trains the network to predict the next word given the previous word and the previous output of the hidden layer but does store no information in the hidden layer that is useful in the future. With a modification to the algorithm, the network can learn what information to store in the hidden layer. The algorithm is then called backpropagation through time algorithm. The idea is to unfold the recurrent neural network for N time steps which can be then seen as deep feedforward network with N hidden layers as depicted in Figure 4.2.

Errors $e_h(t)$ from the hidden layer $s(t)$ are then recursively propagated to the hidden layer $s(t-1)$ from the previous time step as shown in Equation 4.27.

$$e_h(t-r-1) = d_h(e_h(t-r)^T W, t-r-1) \tag{4.27}$$

The weigth matrices U and W are then updated as

$$U(t+1) = U(t) + \sum_{z=0}^{T} w(t-z)e_h(t-z)^T\alpha - U(t)\beta \tag{4.28}$$

$$W(t+1) = W(t) + \sum_{z=0}^{T} s(t-z-1)e_h(t-z)^T\alpha - W(t)\beta \tag{4.29}$$

where T is the number of time steps fpr which the network is unfolded in time.

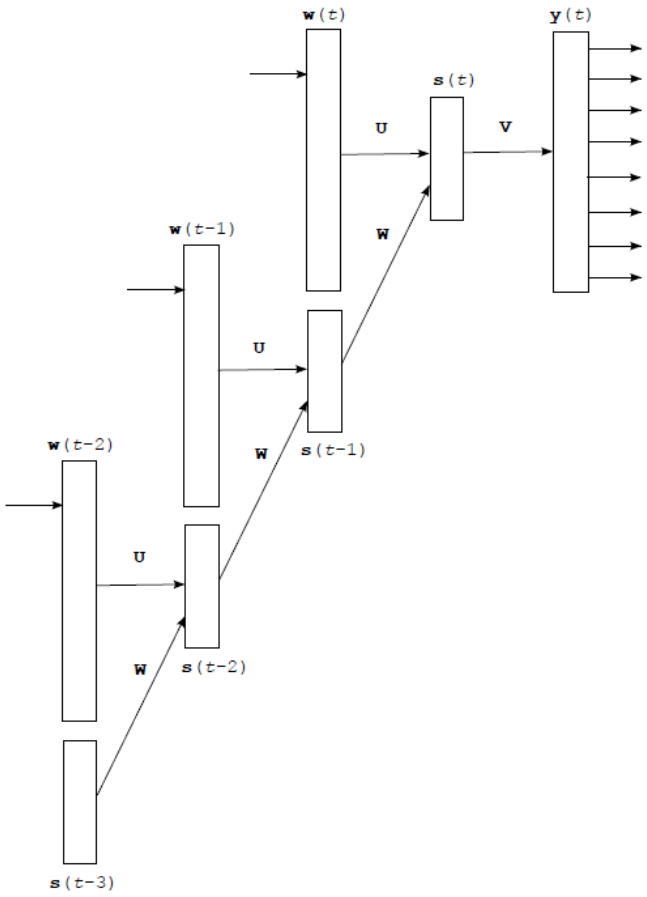


Figure 4.2: Recurrent neural network unfolded as a deep feedforward network, here for 3 time steps back in time. Taken from Mikolov (2012, p.36)

4.5 Combination of Language Modelling Techniques

There are several approaches to combine different language models such as linear interpolation, log-linear interpolation (Klakow 1998), backing-off, maximum entropy models (Berger, V. J. D. Pietra and S. A. D. Pietra 1996; Ronald Rosenfeld 2005). In this research study, linear interpolation is used due to the ease if estimating the necessary parameters.

4.5.1 Linear Language Model Interpolation

Linear interpolation can be used to combine arbitrary many probability distributions given by language models. Linear interpolation of M language models can be performed as

$$Pr(w_i|H) = \sum_{j=1}^{M} \lambda_j Pr_{M_j}(w_i|H) \tag{4.30}$$

where $\sum_{j=1}^{M} \lambda_j = 1$. The weights can be easily determined by optimising the perplexity on a held-out data set. For two models M_1 and M_2, the equation looks like:

$$Pr(w_i|H) = \lambda Pr_{M_1}(w_i|H) + (1 - \lambda) Pr_{M_2}(w_i|H) \tag{4.31}$$

The combined model is guaranteed to have no higher perplexity than any of its components on the held-out data the weights are optimised on. This is because the interpolation factor λ can be zero for individual components.

However, it is difficult to use such an interpolated model directly in the decoder of the speech recogniser because they are usually designed to load a single language model at a time and commonly handle only N-gram based language models. This problem can be partially resolved by the technique described in the next section.

4.5.2 N-best List Rescoring

Stolcke, Konig and Weintraub (1997) have shown that the 1^{st}-best hypothesis produced is not always optimal with regard to the accuracy metric WER. Thus, a more specialised language model can be used to rescore the N-best list of hypotheses produced by the speech recogniser to achieve a higher recognition accuracy. The benefit of rescoring N-best lists is that the speech recogniser itself must not support the used language modelling technique. However, rescoring only reorder or recombine hypotheses but not create new hypotheses.

In this research project, N-best list rescoring is used to measure the performance of the evaluated language modelling techniques. This is done by keeping computing a $N - best$ list of hypotheses using a simple language model M_{weak} and storing the factor of the probability produced by the acoustic model $Pr(A|hyp)$, as outlined in section 1.2, for each of the N hypotheses. This factor is then combined with the a priori probability of the respective hypotheses $Pr_M(hyp)$ given by the language model M used to rescore the N-best list. The new best hypothesis $\hat{hyp}$ can then be determined as

$$\hat{hyp} = \arg\max_{hyp} Pr(A|hyp) Pr_M(hyp). \tag{4.32}$$

Chapter 5

Experiments

In this chapter, the data collection, data generation, the conducted experiments to evaluate the performance of word-based N-gram language models, class-based N-gram language models, recurrent neural network language models and combination of these models as well as the methods for analysing the results are presented.

5.1 Assumptions and Hypothesis

The following hypothesis and assumptions are deducted based on the research studies discussed in chapter 2 and chapter 4.

Assumption

1. A language model which performs better in rescoring N-best hypotheses list also performs better when directly used in the speech recogniser.

Hypotheses

1. Rescoring 1000-best hypotheses lists can improve WER.
2. Rescoring a 1000-best hypotheses list with the language model used to create the it does not improve WER.
3. MKN smoothing outperforms KN smoothing on word-based 3-gram language models used to rescore 1000-best hypotheses lists.

4. RNN based language models, as well as linear combinations of class and word based models, RNN and word-based models, and RNN, class and word-based models improve the 1-best result produced by the speech recogniser when rescoring 1000-best lists.
5. Word based 3-gram language models are outperformed by RNN based language models, as well as linear combinations of class and word-based models, RNN and word-based models, and RNN, class and word-based models when rescoring 1000-best lists.

5.2 Medical Reports Dataset

As described in chapter 3, data needs to be collected to test the above stated hypotheses.

To train the dictionary and the acoustic model, the Verbmobil 1 corpus[1] has been selected. The Verbmobil project, funded by the German Ministry of Science and Technology (BMBF), was carried out from 1993 to 2000 (Wahlster 2000). The Verbmobil corpus contains dialog speech in three languages (English, Japanese, German) in the appointment scheduling task. Statistics of the training part of the German Verbmobil 1 portion of the whole Verbmobil corpus which is used in this research study is shown in Table 5.1.

Set	Sentence	Tokens	Types	Audio [h]
training	12,590	285,168	6,452	30.5

Table 5.1: Verbmobil 1 corpus used for acoustic model training

The audio conditions of the Verbmobil 1 corpus are as recorded in a quiet office environment. This audio conditions are similar to the audio conditions a medical report is recorded in. The included 30.5h of transcribed speech should be sufficient to train a reasonable well performing acoustic

[1] Available from:http://www.phonetik.uni-muenchen.de/Bas/BasVM1eng.html

model such that the speech recognition accuracy is mostly influenced by the trained language model (Pellegrini and Lamel 2008).

As mentioned in chapter 1, the task domain are German medical reports particularly from radiology. The in Dresden, Germany based company Linguwerk GmbH provided about 43,000 medical reports to support this research project. The preprocessing steps performed to use the reports for language modelling have been described in previous works (Lange 2014a,b). The resulting corpus has been divided into a training, development and evaluation portion. The training portion which is further split into sets of different size is used to train the language models. The development set is used to optimise weights and the evaluation set is used to verify that the results produced on the development set did not occur due to overfitting. Statistics of the created development and evaluation set as well as statistics of the training sets with varying size are shown in Table 5.2 and Table 5.3.

Set	Sentence	Tokens	Types	Audio[1] [h]
dev	500	4,337	1,274	1.65
eval	500	4,411	1,284	1.75

Table 5.2: Development and evaluation set created from medical reports

Audio recordings of the development and evaluation set are necessary to produce the N-best hypotheses lists with the speech recogniser. Thus, both sets were recorded by 2 male speakers in their early-twenties resulting in a total of 1000 utterances for each set. The audio was recorded as 16kHz, 16bit mono channel audio with the microphone of a Logitech G230 headset in a quiet office room to match the properties of the audio data used to train the acoustic model.

5.3 Experimental Setup

The Kaldi Speech Recognition Toolkit (Povey et al. 2011) is used to train the acoustic model. In previous performed comparison of freely available

[1]Total audio data in hours produced by both speakers

Sentence	Tokens	Types	Singletons	OOV tokens (dev set)		OOV tokens (eval set)	
				n	%	n	%
1k	8,468	1,872	1,147	622	14.34	641	14.53
2k	16,869	2,854	1,639	432	9.96	453	10.27
10k	83,504	6,572	3,198	187	4.34	171	3.88
20k	167,833	9,279	4,342	128	2.95	108	2.45
50k	421,235	14,173	6,450	73	1.68	64	1.45
100k	848,297	19,330	8,474	53	1.22	43	0.97
200k	1,677,484	26,153	11,382	33	0.76	27	0.61
500k	4,200,089	38,893	16,604	23	0.53	18	0.41
795k	6,677,442	47,249	19,912	14	0.32	12	0.27

Table 5.3: Training sets of different size created from medical reports

open source speech recognisers (Gaida et al. 2014), Kaldi has shown to outperform the other compared recognition toolkits and provides the most advanced techniques.

The produced acoustic model is a p-norm deep neural network model (Zhang et al. 2014) with 4 hidden layers, a dimensionality of the input/output layer of 2400/300 and was trained in 12 epochs. The used features are Mel Frequency Cepstral Coefficients (MFCC) with 13 cepstra spliced over 4 frames in each direction. Furthermore, a Linear Discriminant Analysis (LDA) transform, a Maximum Likelihood Linear Transform (MLLT) and Feature Space Maximum Likelihood Linear Regression (fMLLR) are applied to the features which results in a 40-dimensional feature vector. Additionally, an iVector is supplied to the network containing the properties of the speaker. The iVector is estimated on the full utterance and can be carried forward to the next utterance of the same speaker. The in Kaldi included decoder for the above described acoustic model is used to produce the N-best hypotheses lists.

The SRILM Toolkit (S. C. Martin, Liermann and Ney 1997) is used to generate the word-based N-gram language models, the class mapping for the class-based N-gram language models and the combinations of non-RNN

based language models. Furthermore, it is also used to rescore the N-best hypotheses lists.

The Kylm Toolkit is used to train class-based 3-gram language models given the class mapping produced with the SRILM Toolkit. SRILM only supports training of

The RNNLM Toolkit is used to train the RNN based language models and combinations including a RNN language model.

The data-driven grapheme-to-phoneme converter Sequitur G2P (Bisani and Ney 2008) developed at the RWTH Aachen University was used to create the required dictionary from the vocabulary of the medical reports. The Sequitur G2P model was trained on the dictionary included in the Verbmobil 1 corpus.

5.4 Experimental Design

The language models shown in Table 5.4 are created for each of the training sets shown in Table 5.3.

Name	Description
mKN	Word based 3-gram model with mKN smoothing
KN	Word based 3-gram model with KN smoothing
cX	Class based 3-gram model with X classes
RNN	RNN based model

Table 5.4: Trained language model types

Class-based models are trained with $100, 200, \ldots, 1500$ classes as well as 2000 classes for the two largest training sets (500k and 795k). The RNN models are all trained with a hidden layer dimension of 100 and are unfolded into 4 time steps representing a deep neural network with 4 hidden layers. The output layer is factorised into 100 classes based on word occurrence frequencies to speed up the training process.

In an initial experiment, the mKN model is used to create N-best hypotheses lists with the default ratio between acoustic and language model influence of 1 : 10. This ratio is then optimised for each of the various training sizes on the development data. N is chosen to be 1, 10, 50, 100, 200, 500 and 1000. In the following experiments 1000-best hypotheses lists produced with the optimised ratio between acoustic and language model are rescored. The WER of the 1-best hypothesis lists produced with the mKN model as well as the rescoring result of the best word based model are used as baseline.

In the second experiment, the perplexity with respect to the development set of the models included in hypotheses 3 - 5 is computed for quick evaluation and finding the optimal weights for the models created using linear interpolation.

To test the first hypothesis that rescoring 1000-best lists can improve the baseline WER, the WER of an oracle which always knows which hypothesis results in the least number of errors is computed. Afterwards, the rescoring of the 1000-best lists of the development set is performed to produce the data necessary to test hypotheses 2-5. In a last experiment, 1000-best lists of the evaluation set are rescored with models produced from the 50k training set to ensure that the results seen in the development data are not due to overfitting. To ensure statistical significance, null and alternative hypotheses will be formulated and tested for a significance level of $\alpha = 0.05$.

Chapter 6

Results

6.1 Increase Training Set Size

In Figure 6.1, the WER of the 1-best hypotheses list produced with the default acoustic-scale parameter of 0.1 is shown. It can be seen that the WER rapidly decreases when the training set size is increased from 1k to 100k. Afterwards, only a slight decrease in WER can be observed when the training set sized is further increased up to 795k.

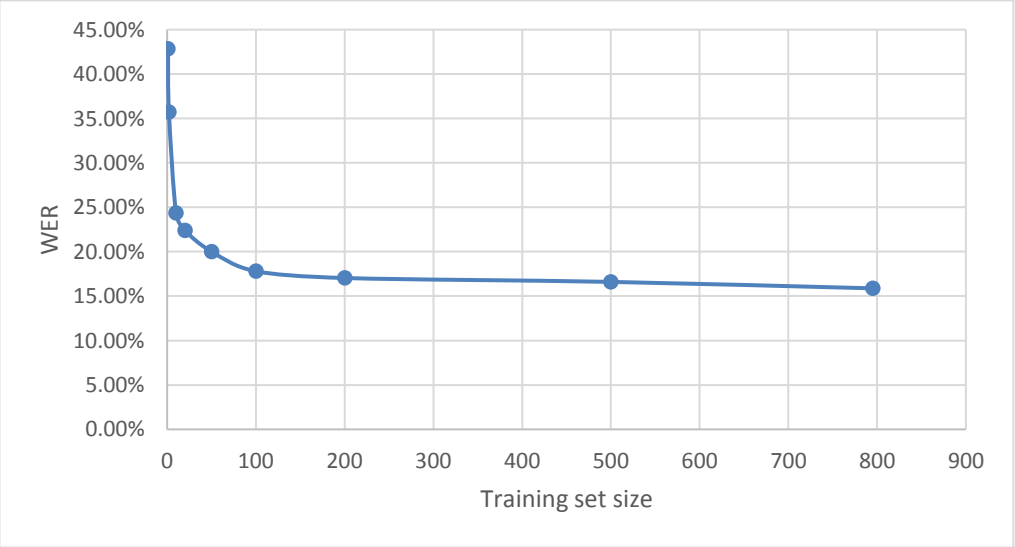


Figure 6.1: 1-best list WER with default acoustic-scale of 0.1

The WER results produced by optimising the acoustic-scale parameter over the ranges from 0.01 to 0.1 are presented in Figure 6.2. The graphs for all

training sets have the same parabolic shape with a WER minimum at either 0.04 or 0.05. These values correspond to a ratio between the acoustic-cost and the language model cost used by the decoder to determine the best hypothesis of 1:25 or 1:20 respectively. The graphs indicate that the optimal value would be somewhere between 0.04 and 0.05 but closer to 0.04 because the distance between the 0.04 and 0.05 WER value is minimal when 0.05 is the optimal configuration.

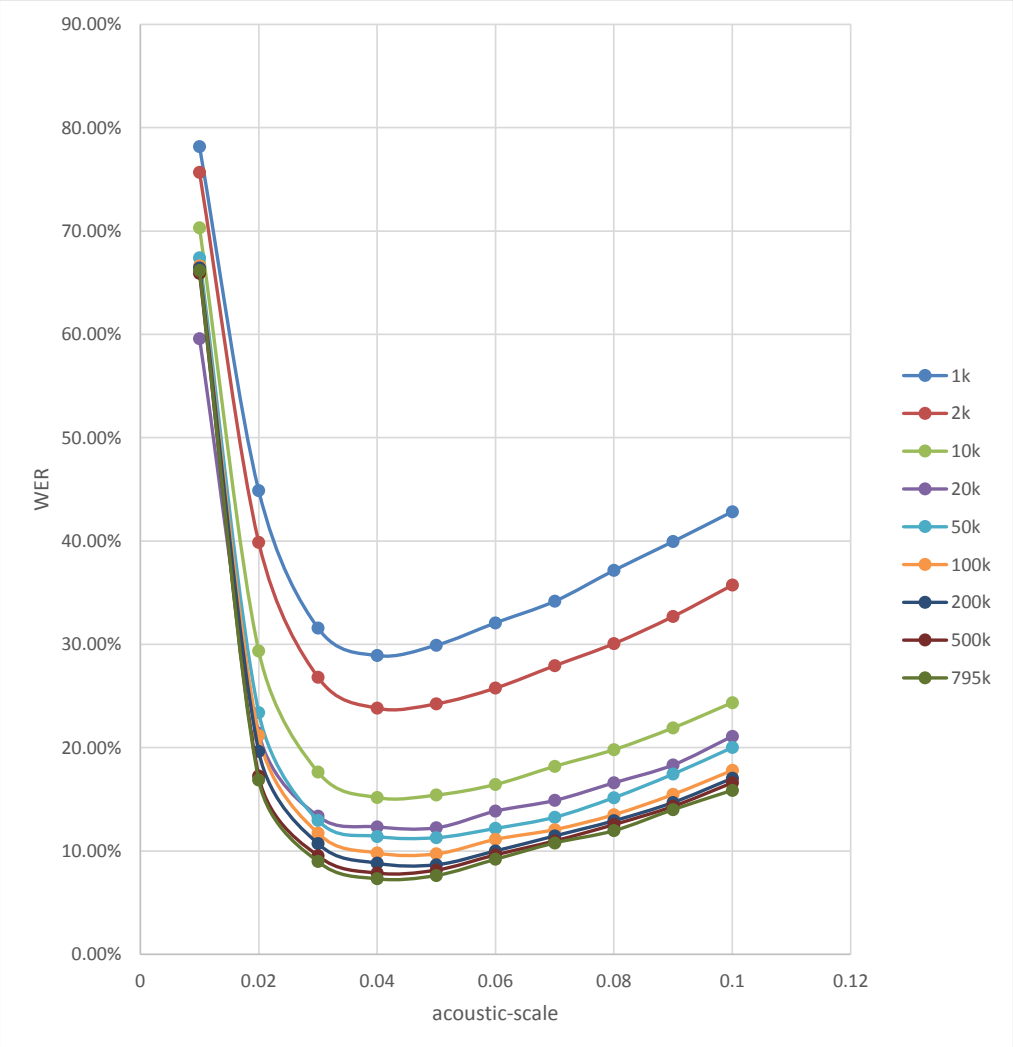


Figure 6.2: 1-best list WER with varying acoustic-scale

The results of the default and the optimised acoustic-scale parameter are shown together with the absolute WER reduction of the optimisation in Figure 6.3. It can be seen that the absolute WER reduction is almost constant at around 8.5% for the training sets 10k and greater. The acoustic-scale optimisation does only produce higher WER improvements of 14.5% and 11.5% for the 1k and 2k training set.

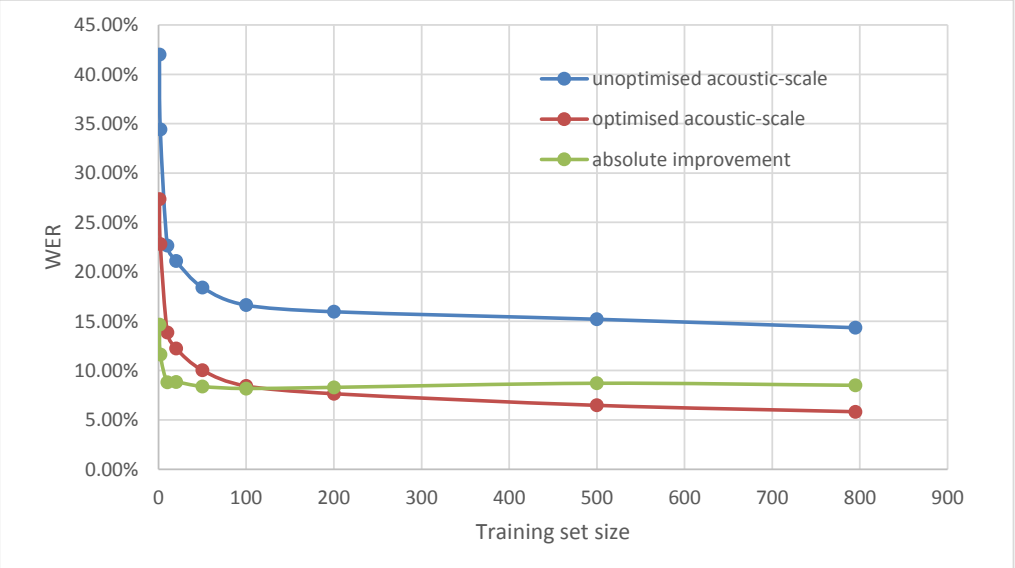


Figure 6.3: 1-best list WER for default and tuned acoustic-scale parameter

The above presented results indicate that the optimisation of the ratio is caused by the acoustic model and not the language model. If the language model would cause the improvement, it would be expected that better language models result in a lower acoustic-scale parameter because that would emphasis the language model. Although, the language models trained on larger training sets produce lower WER, they show the have the same acoustic-scale parameter like the models trained from smaller training sets.

6.2 Perplexity Experiments

The results of the perplexity experiments are presented in this section.

6.2.1 Word-Based Models

Table 6.1 shows the perplexity values on the development set for both word-based models with mKN and KN smoothing. The perplexity of the KN smoothed model is always lower than the perplexity of the mKN model. This results indicate that the KN smoothing performs better on the data used in this research project. Based on this results, the word-based KN model is as comparison in further experiments and for linear interpolation with other language modelling techniques.

Model	1k	2k	10k	20k	50k	100k	200k	500k	795k
mKN	22.6188	21.8303	16.9974	15.0796	13.4678	12.0966	11.295	10.164	9.6876
KN	18.6077	18.4153	15.1031	13.6647	12.623	11.2691	10.5773	9.67501	9.30097

Table 6.1: Perplexity values of the mKN and KN smoothed word-based models on the development set

6.2.2 Class-Based Models

The perplexity results for the class-based language models are presented in Table 6.2. All class models produce higher perplexity than the corresponding KN smoothed word-based model. Furthermore, the perplexity decreases when the number of classes is increased. This is expected behaviour since a class model with as many classes as there are words in the vocabulary is identical to a word-based language model.

6.2.3 RNN Based Models

In Table 6.3, showing the perplexity results for the RNN based models, it can be seen that the RNN based models produce higher perplexity values for the small training data sets up to 20k than the KN smoothed word-based model and lower values for the larger training sets. The large value

Model	1k	2k	10k	20k	50k	100k	200k	500k	795k
KN	18.6077	18.4153	15.1031	13.6647	12.623	11.2691	10.5773	9.67501	9.30097
c100	32.2804	36.3968	37.5779	37.3486	37.8989	37.5908	37.6216	39.6971	40.1869
c200	26.3249	27.5918	27.3364	26.6431	26.1406	25.7131	25.0159	26.2893	26.2259
c300	23.3949	24.548	23.6336	22.7584	22.0149	21.249	20.8278	21.3769	21.3206
c400	22.526	23.3107	21.7592	20.5967	19.4605	18.7646	18.2582	18.6076	18.4223
c500	21.2219	22.422	20.1764	19.069	17.9714	17.1152	16.7037	16.9434	16.8607
c600	20.354	21.6302	19.3313	17.8915	17.2236	16.2324	15.406	15.6937	15.3786
c700	20.0813	21.0521	18.5625	17.4858	16.3816	15.3973	14.686	15.0433	14.7458
c800	19.6405	20.4764	18.1338	16.8323	16.0046	14.8008	14.1089	14.096	14.0315
c900	19.3597	19.9683	17.707	16.378	15.6561	14.1876	13.556	13.651	13.4471
c1000	19.1705	19.7615	17.2254	16.1019	14.9941	13.6986	13.2763	13.3367	13.0649
c1100	18.993	19.3952	17.072	15.8281	14.7576	13.5408	13.1128	12.978	12.7544
c1200	18.8859	19.2902	16.9035	15.4951	14.5689	13.3501	12.8605	12.6838	12.6002
c1300	18.6033	18.983	16.7217	15.4314	14.3624	13.092	12.4829	12.498	12.2921
c1400	18.5636	18.899	16.5364	15.2683	14.1455	12.9988	12.3725	12.1934	12.0385
c1500	18.445	18.742	16.3225	15	13.977	12.8058	12.2996	12.0071	11.8142
c2000	18.6077	—	—	—	—	—	—	11.3857	11.2427

Table 6.2: Perplexity values of the class-based language models on the development set

for the 795k training set is most likely an anomaly caused by not tuning the parameters of the RNN model. As mentioned in section 5.4, all RNN models have been trained with the same parameterisation which seems to be not optimal. perplexity values for the RNN based language models are shown in Table 6.3 The results indicate that the RNN based language models perform better with more training data.

Model	1k	2k	10k	20k	50k	100k	200k	500k	795k
KN	18.6077	18.4153	15.1031	13.6647	12.623	11.2691	10.5773	9.67501	9.30097
RNN	19.92673	20.151181	15.992711	14.318321	12.571755	10.936137	10.076213	9.311189	27.131566

Table 6.3: Perplexity values of the RNN based language models on the development set

6.2.4 Combination of Language Models Performance

The results produced by the linear combination of the class-based models with the KN smoothed word-based model are shown in Table 6.4. The interpolation factor for the best combination applied to the class model is presented in Table 6.5. It can be seen that the linear combination of almost

all class models with the KN smoothed word-based model can outperform the word-based component.

Model	1k	2k	10k	20k	50k	100k	200k	500k	795k
KN	18.6077	18.4153	15.1031	13.6647	12.623	11.2691	10.5773	9.67501	9.30097
c100+KN	18.1267	18.0067	14.9028	13.5825	12.5197	11.2707	10.5992	9.74009	9.36577
c200+KN	18.381	17.9677	14.8354	13.4896	12.4287	11.2019	10.5465	9.70432	9.35045
c300+KN	18.348	17.9724	14.8362	13.471	12.4028	11.159	10.4922	9.65735	9.29735
c400+KN	18.4487	18.1472	14.8431	13.4235	12.304	11.1247	10.4571	9.62341	9.2646
c500+KN	18.4361	18.1807	14.8034	13.4258	12.2604	11.0733	10.4518	9.60344	9.26191
c600+KN	18.4052	18.1887	14.857	13.3526	12.3554	11.0909	10.4195	9.60438	9.22672
c700+KN	18.4133	18.2323	14.8421	13.4266	12.2662	11.0797	10.3967	9.61704	9.26088
c800+KN	18.3856	18.2151	14.9081	13.3981	12.3188	11.0627	10.4075	9.58273	9.24106
c900+KN	18.3361	18.1831	14.8868	13.4077	12.3735	11.0183	10.3692	9.58818	9.20789
c1000+KN	18.3574	18.2555	14.8419	13.4558	12.3054	10.996	10.3598	9.59766	9.23446
c1100+KN	18.3493	18.2168	14.893	13.4342	12.3356	11.0473	10.4011	9.59361	9.22609
c1200+KN	18.3055	18.2512	14.9222	13.4048	12.3644	11.0317	10.4291	9.59929	9.24026
c1300+KN	18.244	18.1785	14.9463	13.4556	12.3453	11.0177	10.361	9.59718	9.22357
c1400+KN	18.2867	18.1759	14.9305	13.4522	12.3024	11.024	10.3867	9.57185	9.22858
c1500+KN	18.2832	18.1695	14.9177	13.42	12.319	11.0191	10.4087	9.57737	9.2425
c2000+KN	—	—	—	—	—	—	—	9.56727	9.21011

Table 6.4: Perplexity values of the combination of class- and word-based models on the development set

However, the interpolation factors show that the class-based model makes up for only a small part in the resulting combination. The interpolation factors of related to two largest training data sets indicate that the chosen number of classes has not been optimal for these training sets. A larger number of classes would most likely be better for the 500k and 795k training set due to a larger vocabulary. The general trend that can be observed is that with increasing training set size, and thus increasing vocabulary size, higher number of classes perform better.

	1k	2k	10k	20k	50k	100k	200k	500k	795k
Interpolation factors	0.1	0.1	0.1	0.15	0.15	0.15	0.15	0.1	0.05

Table 6.5: Interpolation factors of the class-based model

The perplexity values of the linear combinations including a RNN based model are presented in Table 6.6. Table 6.7 shows the related interpolation factors applied to the RNN component in the models. When compared with Table 6.4, it can be observed that the combinations including a RNN

based model always outperform the individual components. Furthermore, the linear interpolation of the RNN based model with the best interpolation between class and word-based model produce the lowest perplexity values.

Model	1k	2k	10k	20k	50k	100k	200k	500k	795k
KN	18.6077	18.4153	15.1031	13.6647	12.623	11.2691	10.5773	9.67501	9.30097
RNN	19.92673	20.151181	15.992711	14.318321	12.571755	10.936137	10.076213	9.311189	27.131566
RNN+KN	17.877535	17.603723	14.169108	12.727786	11.415832	10.129256	9.37072	8.683796	9.196912
RNN+(cX+KN)	17.674399	17.420681	14.033653	12.550955	11.262467	10.018137	9.299469	8.668858	9.132881

Table 6.6: Perplexity values of the combinations including a RNN based model on the development set

The in comparison to the other very low interpolation factor for the 795k training set reflects the weak performance of the RNN component for this training set discussed above. The fact that the linear combination of the RNN based model with the word-based model of the 795k model still outperforms the best linear combination of class and word-based model is most likely because the evaluated number of classes for are not large enough for the 795k training set as explained above.

Model	1k	2k	10k	20k	50k	100k	200k	500k	795k
RNN+KN	0.4	0.4	0.45	0.5	0.55	0.6	0.65	0.6	0.1
RNN+(cX+KN)	0.35	0.35	0.45	0.45	0.55	0.55	0.6	0.6	0.05

Table 6.7: Interpolation factors of the RNN based model

Although it seems that the class and RNN based models have not been optimally tuned for the largest training set, it will still be included in the results of further experiments for completeness.

6.3 N-Best List Rescoring

In this section, the results of the N-best hypotheses list rescoring experiments are presented.

6.3.1 Oracle Word Error Rate

The oracle WER with varying size of the N-best hypotheses list for the different training sets is shown in Figure 6.4.

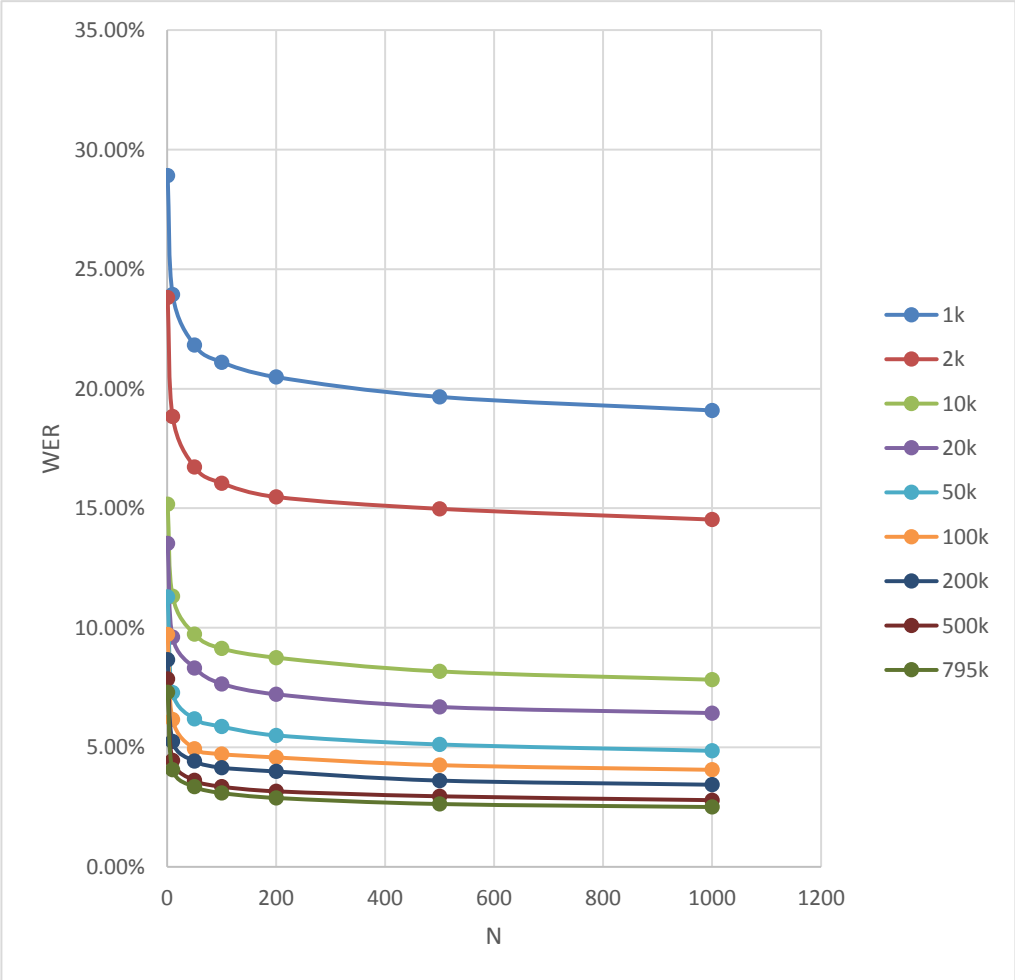


Figure 6.4: Oracle WER with varying N

In Figure 6.4, showing the oracle WER over the size of the N-best hypotheses list for the different training sets, it can be observed that the oracle WER decreases in the same way for all training sets with increasing N. The oracle WER improvement produced when increasing N decreases for larger N. This results show that it is theoretically possible to improve the baseline

by rescoring N-best hypotheses lists. Furthermore, the suggest that the benefit of rescoring very large N-best hypotheses lists is negligible.

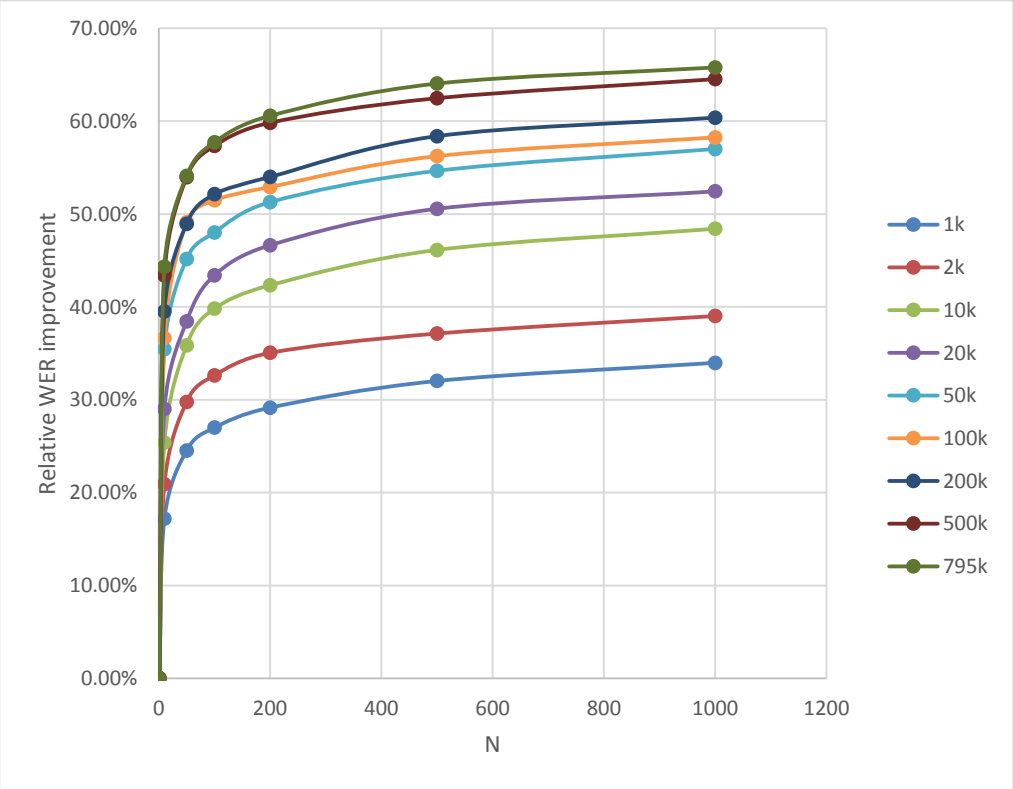
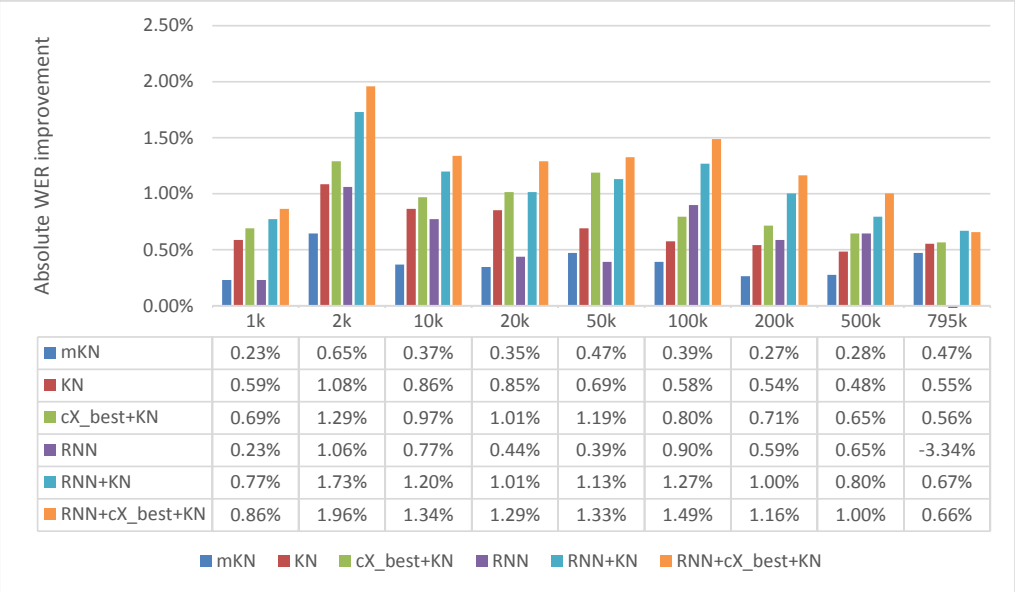


Figure 6.5: Relative WER improvement of the oracle over the baseline with varying N

The theoretically possible relative WER improvement over the baseline is shown in Figure 6.5. While the related absolute WER improvements of the oracle range from 10% to 5% for the 1k to the 795k model on 1000-best hypotheses lists, the relative improvement shows the exact opposite relationship. The relative improvement is larger for larger training sets. This is most likely because the N-best hypotheses list of the larger training sets have been produced with a better performing language model.

6.3.2 Development Experiments

The absolute WER improvement achieved when rescoring 1000-best hypotheses lists with the evaluated language modelling techniques is shown in Figure 6.6.



	1k	2k	10k	20k	50k	100k	200k	500k	795k
mKN	0.23%	0.65%	0.37%	0.35%	0.47%	0.39%	0.27%	0.28%	0.47%
KN	0.59%	1.08%	0.86%	0.85%	0.69%	0.58%	0.54%	0.48%	0.55%
cX_best+KN	0.69%	1.29%	0.97%	1.01%	1.19%	0.80%	0.71%	0.65%	0.56%
RNN	0.23%	1.06%	0.77%	0.44%	0.39%	0.90%	0.59%	0.65%	-3.34%
RNN+KN	0.77%	1.73%	1.20%	1.01%	1.13%	1.27%	1.00%	0.80%	0.67%
RNN+cX_best+KN	0.86%	1.96%	1.34%	1.29%	1.33%	1.49%	1.16%	1.00%	0.66%

Figure 6.6: Absolute WER improvement over the 1-best hypotheses list

In section 5.4, it was assumed that rescoring a N-best hypotheses list with the model it was created with does not change the WER. However, it can be observed that even rescoring with the mKN smoothed word-based model yields a small improvement over the baseline. The fact that Kaldi does not only use the language model probabilities but a combination of it with pronunciation and word transition probabilities to compute the best hypothesis is most likely the reason for this improvement.

As in the perplexity experiments, the KN smoothed word-based models outperform the mKN smoothed models for all training data sizes.Likewise, the RNN based models show a very similar trend as indicated by the perplexity values. The only difference to the perplexity values is that the RNN models start to become better for training set sizes greater than 100k instead

of 50k. Furthermore, the combinations also follow the indications of the perplexity. The best linear combination of class-based and KN smoothed word-based model always outperforms the word-based component. Linear combinations containing a RNN component give better absolute WER improvements than all other techniques with the combination of RNN, class and word-based model performing best.

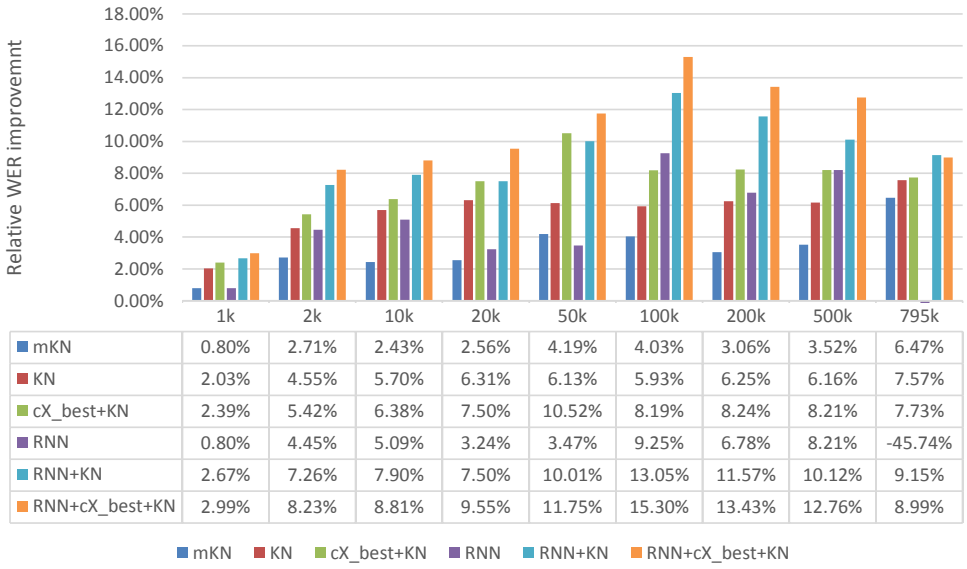
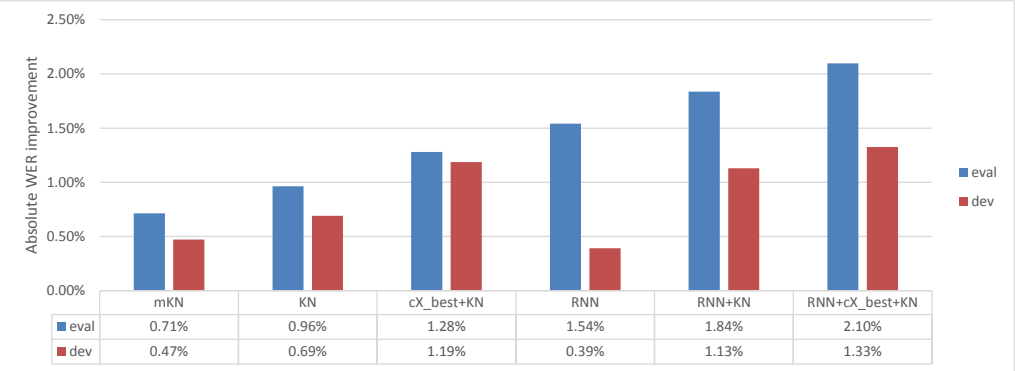


Figure 6.7: Relative WER improvement over the 1-best hypotheses list

Figure 6.7 shows the relative WER improvement of the evaluated language models. It can be observed that the relative improvement of the KN smoothed word-based model and its linear combination with a class model seems to stay constant after the training set size has been increased past 10k and 20k respectively. However, the relative improvement of the RNN language model seems to increase with increasing training data size. The results indicate that the relative WER improvement achieved by linear combinations including a RNN component peaks at a training size of 100k. This finding suggests that rescoring with a linear combination of a RNN, class and word-based language model is most effective at a training set size of 100k.

6.3.3 Evaluation Experiment

Figure 6.8 compares the absolute WER improvements for the 50k training set on the development and evaluation data. The data shows that the relationship between between the word-based models and the best linear combination of the class model with the KN smoothed word-based model is the same for the development and the evaluation data. The same is true for the relationship between the models containing a RNN component. However, the second group seems to perform on the evaluation data than on the development data. This seems to be cause by the RNN model.



	mKN	KN	cX_best+KN	RNN	RNN+KN	RNN+cX_best+KN
eval	0.71%	0.96%	1.28%	1.54%	1.84%	2.10%
dev	0.47%	0.69%	1.19%	0.39%	1.13%	1.33%

Figure 6.8: Absolute WER improvement on the development and evaluation data for the 50k training set

The same observations as made in Figure 6.8 can be made in Figure 6.9 showing the relative WER improvement for the 50k training set on the development and evaluation data.

6.4 Analysis of the Results

Significance testing of the results of the oracle experiments, the rescoring experiments for varying training data size on the development set and of the results produced by the rescoring experiments on the evaluation set using the 50k training set is performed in this section.

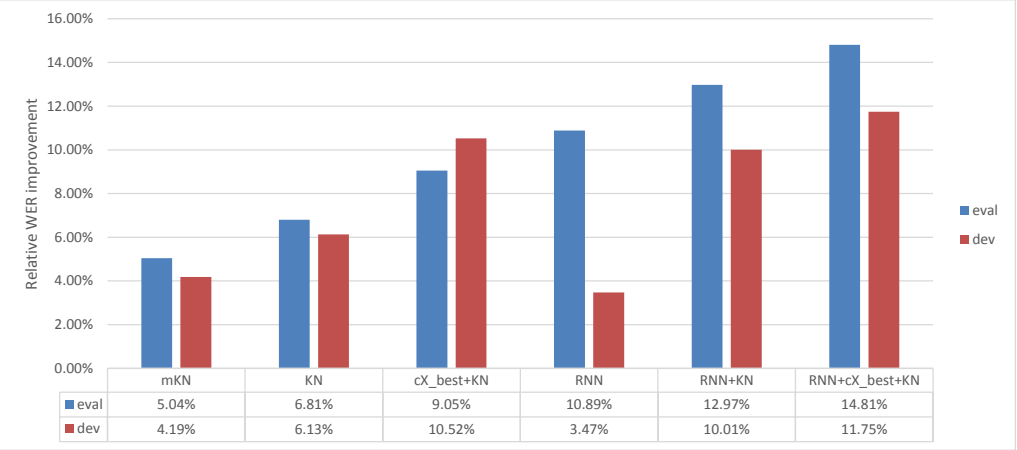


Figure 6.9: Relative WER improvement on the development and evaluation data for the 50k training set

Since two proportions p_1 and p_2 corresponding to the produced WERs are compared, the z-test is used to test statistical significance of the results. The null and alternative hypothesis pairs are of the form

$$\begin{aligned} H_0 &: p_1 - p_2 \leq 0 \\ H_1 &: p_1 - p_2 > 0 \end{aligned} \tag{6.1}$$

to test whether p_2 is statistically significantly smaller than p_1. The computed z-score has to be greater than 1.644853 to be able to reject H_o at a significance level of $\alpha = 0.05$ for the one-tailed test shown in Equation 6.1.

The results of the z-test presented in Table 6.8 show that the WER of the oracle on 1000-best lists is statistically significantly lower than the baseline produced by the speech recogniser across all training set sizes.

Moreover, the results show that rescoring 1000-best lists with the mKN smoothed and KN smoothed word-based 3-gram language models does not produce statistically significant improvement for most of the training set sizes. The mKN model does not produce any statistical significant improvement as would have been expected since it was used to create the 1000-best lists. The KN model only produces statistically significant improvement for the 2k and 20k training set when applied to the development data and for

Null hypothesis			z-Score									
p_1	p_2	Tails	1k	2k	10k	20k	50k	100k	200k	500k	795k	eval
1-best	Oracle	1	15.2	15.4	15.2	15.6	15.6	14.7	14.5	14.9	14.7	—
1-best	mKN	1	0.34	0.46	0.68	0.67	0.99	0.88	0.62	0.68	1.21	1.38
1-best	KN	1	0.86	1.69	1.61	1.67	1.46	1.30	1.29	1.20	1.43	1.86
1-best	cX+KN	1	1.01	2.02	1.80	1.99	2.53	1.80	1.71	1.61	1.46	2.49
1-best	RNN	1	0.34	1.65	1.43	0.85	0.82	2.04	1.40	1.61	-7.70	3.01
1-best	RNN+KN	1	1.13	2.71	2.24	1.99	2.40	2.91	2.41	1.99	1.73	3.60
1-best	RNN+(cX+KN)	1	1.26	3.07	2.50	2.54	2.83	3.43	2.81	2.53	1.70	4.13
mKN	KN	1	0.52	0.69	0.93	1.00	0.47	0.42	0.66	0.52	0.21	0.49
KN	cX+KN	1	0.15	0.33	0.20	0.32	1.07	0.50	0.42	0.41	0.03	0.63
KN	RNN	1	-0.52	-0.04	-0.17	-0.82	-0.64	0.74	0.11	0.41	-9.10	1.15
KN	RNN+KN	1	0.27	1.02	0.63	0.32	0.95	1.61	1.13	0.79	0.30	1.74
KN	RNN+(cX+KN)	1	0.41	1.39	0.90	0.87	1.38	2.13	1.53	1.33	0.27	2.27

Table 6.8: z-scores for the tested hypotheses (statistical significant results highlighed)

the 50k training set when applied to the evaluation data which most likely are due to the noisiness of the WER metric.

Results similar to the KN model are shown for the RNN based models. The overall trend is that the improvements over the baseline produced by the speech recogniser are not statistically significant. The RNN based model shows significant improvement over the speech recogniser baseline only for the 2k and 200k training set as well as the 50k training set when used for evaluation.

Furthermore, the results for the models created by linearly interpolating a class-based 3-gram model with the KN smoothed word-based 3-gram model, a RNN based model with the same word-based model and interpolating all three of these models show mostly statistically significant improvement over the baseline produced by the speech recogniser. The only exception is the 1k training set for which all 3 combinations do not statistically significant improvement and the 500k and 795k training set for the combination of the class and word-based model. This is most like due to the high out of vocabulary rate (OOV) rate of 14.34% for the 1k training set on the development data and that the class number is most likely not optimal for the 500k and 795k training set as mentioned above.

Finally, no statistical significant improvements of the advanced techniques

can be reported when compared to the rescoring results of the KN model. However, a trend in the z-scores can be observed that for all training set sizes the order from the lowest to highest z-score is mKN, KN, cX+KN, RNN+KN, RNN+(cX+KN). Only the RNN model has no clear place in this ranking.

Chapter 7

Discussion

The research question asked in this study is

> "Can advanced language modelling techniques particularly RNN and class-based language models and their combinations increase the speech recognition accuracy when trained on only limited amounts of training data consisting of German medical reports particularly from radiology when compared to the standard word-based language model?" .

The results presented and analysed in chapter 6 show that the research question can be partially answered with *yes*. While the isolated usage of the advanced modelling techniques shows no improvement over the standard word-based 3-gram language model with mKN smoothing used in the speech recogniser, the evaluated linear combinations of the advanced techniques produce statistically significant improvement over the speech recogniser baseline when used to rescore 1000-best lists for the 2k and larger training sets. It can be expected that the techniques could perform even better when they are directly used in the speech recogniser.

While some of the presented findings agree with the results presented in the reviewed literature, others do not. The findings that class-based language models on their own do not perform better than word-based language models (Brown et al. 1992; S. Martin, Liermann and Ney 1998) but when interpolated with a word-based language model outperform it (S. Martin,

Liermann and Ney 1998) are backed up by the results of this study. Furthermore, the idea to jointly evaluate language models to achieve better performance proposed by J. T. Goodman (2001) could be applied successfully.

However, the trend that the improvement, advanced language modelling techniques provide, decreases with more training data (J. T. Goodman 2001) with exception of RNN language models for which the improvement they provide should increase with more training data cannot be backed up by the results of this study. The results shown in Figure 6.7, suggest that the improvement achieved by the KN smoothed word-based model and its interpolation with the class-based model stays constant for training set sizes greater than 10k and 20k respectively. When combined with the observation that mKN smoothing seems to increase in efficiency with increasing training data size, it is most likely that with bigger data sets we could observe the decrease in performance described by J. T. Goodman (2001). In addition, this most likely also explains why mKN smoothing does not outperform KN smoothing as suggested by Chen and J. Goodman (1999). Based on the trends observable in the results produced in this study, it is most likely that with greater training data sets mKN smoothing is better than KN smoothing.

Hnatkowska and Sas (2008) performed a similar speech recognition experiment in the medical domain. However, the language was Polish and the used speech recogniser HTK and not Kalid. They achieved a WER of 16%. We achieve a similar performance with the 10k training data set which is about the size of 500 of the medical reports as shown in Table 7.1.

Training set	Reports	OOV rate	$WER_{1\text{-best}}$	$WER_{RNN+cX+KN}$
1k	50	14.34%	28.91%	28.05%
2k	100	9.96%	23.82%	21.86%
10k	500	4.31%	15.17%	13.83%
20k	1000	2.95%	13.52%	12.23%
50k	2500	1.68%	11.29%	9.96%
100k	5000	1.22%	9.72%	8.23%
200k	10000	0.76%	8.67%	7.51%
500k	25000	0.53%	7.86%	6.86%
795k	40000	0.32%	7.31%	6.65%

Table 7.1: Absolute WER results for different training data sizes

Chapter 8

Conclusions and Future Work

The experiment results proved the thesis that advanced language modelling techniques can be effectively used to prototype a language model for transcribing German medical reports particularly from radiology with small training data sets. The results show that at least around 500 medical reports should be used to ensure a OOV rate smaller than 4.31% and a WER of lower than 15%. Then, the advanced language modelling techniques provided the same improvement as using double the amounts of training data.

In future work, we would like to compare the results obtained by rescoring 1000-best lists with the results achievable by directly applying the advanced language modelling techniques in the speech recogniser. Furthermore, we would like to prove the assumption that larger, more general language model based on a web corpus can not be used as effectively as small amounts of in-domain training data to prototype a language model in this domain.

References

Allison, B., Guthrie, D. & Guthrie, L. (2006). Another Look at the Data Sparsity Problem. In *Proc. of the TSD*. Brno, Czech Republic.

Bahl, L. R., Jelinek, F. & Mercer, R. L. (1983). A Maximum Likelihood Approach to Continuous Speech Recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, *PAMI-5*(2), p.179–190.

Baker, J. (1975). The DRAGON system–An overview. *IEEE Trans. on Acoustics, Speech and Signal Processing*, *23*(1), p.24–29.

Baroni, M., Bernardini, S., Ferraresi, A. & Zanchetta, E. (2009). The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, *43*(3), p.209–226.

Baroni, M. & Ueyama, M. (2006). Building general-and special-purpose corpora by web crawling. In *Proc. of the NIJL international symposium*. Tokyo, Japan.

Bengio, Y., Ducharme, R., Vincent, P. & Janvin, C. (2003). A neural probabilistic language model. *The Journal of Machine Learning Research*, *3*, p.1137–1155.

Berger, A. L., Pietra, V. J. D. & Pietra, S. A. D. (1996). A maximum entropy approach to natural language processing. *Computational Linguistics*, *22*(1), p.39–71.

Biemann, C., Bildhauer, F., Evert, S. et. al. (2013). Scalable construction of high-quality web corpora. *Journal for Language Technology and Computational Linguistics*, *28*(2), p.23–60.

Bisani, M. & Ney, H. (2008). Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, *50*(5), p.434–451.

Brants, T. & Franz, A. (2006). Web 1T 5-gram Version 1 LDC2006T13. Web Download. Philadelphia, USA: Linguistic Data Consortium.

Brants, T., Popat, A. C., Xu, P. et. al. (2007). Large language models in machine translation. In *Proc.. of the EMNLP*. Prague, Czech Republic.

Brown, P. F., deSouza, P. V., Mercer, R. L. et. al. (1992). Class-based N-gram Models of Natural Language. *Computational Linguistics*, *18*(4), p.467–479.

Brun, A., Langlois, D. & Smaïli, K. (2007). Improving language models by using distant information. In *Proc. of the ISSPA*. Sharjah, United Arab Emirates.

Chen, S. F., Beeferman, D. & Rosenfeld, R. (1998). Evaluation metrics for language models. In *Proc. of the DARPA Broadcast News Transcription and Understanding Workshop*. Lansdowne, USA.

Chen, S. F. & Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, *13*(4), p.359–393.

Creswell, J. W. (2013). *Research design: Qualitative, quantitative, and mixed methods approaches* (4th Ed.). Thousand Oaks, USA: SAGE Publications, Inc.

De Groc, C. (2011). Babouk: Focused Web Crawling for Corpus Compilation and Automatic Terminology Extraction. In *Proc. of the IAT*. Lyon, France.

Dean, J. & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM*, *51*(1), p.107–113.

Gaida, C., Lange, P., Petrick, R. et. al. (2014). *Comparing Open-Source Speech Recognition Toolkits*. DHBW Stuttgart. Technical Report of the Project OASIS.

Goodman, J. T. (2001). A bit of progress in language modeling. *Computer Speech & Language*, *15*(4), p.403–434.

Graff, D. & Cieri, C. (2003). English Gigaword LDC2003T05. Web Download. Philadelphia, USA: Linguistic Data Consortium.

Guthrie, D., Allison, B., Liu, W. et. al. (2006). A Closer Look at Skip-gram Modelling. In *Proc. of the LREC*. Genoa, Italy.

Guthrie, D., Guthrie, L. & Wilks, Y. (2009). What is a "full statistical model" of a language and are there short cuts to it. In D. Hlaváčková, A. Horák, K. Osolsobě & P. Rychlý (Eds.), *After Half a Century of Slavonic Natural Language Processing* (p.45–56). Brno, Czech Republic: Masaryk University.

Hnatkowska, B. & Sas, J. (2008). Application of automatic speech recognition to medical reports spoken in Polish. *Journal of Medical Informatics and Technologies*, *12*, p.223–229.

Huang, X., Alleva, F., Hon, H.-W. et. al. (1993). The SPHINX-II speech recognition system: an overview. *Computer Speech & Language*, *7*(2), p.137–148.

Hyde, K. F. (2000). Recognising deductive processes in qualitative research. *Qualitative Market Research: An International Journal*, *3*(2), p.82–90.

Iyer, R., Ostendorf, M. & Meteer, M. (1997). Analyzing and predicting language model improvements. In *Proc. of the ASRU*. Santa Barbara, USA.

Jeffreys, H. (1948). *Theory of Probability* (2nd Ed.). Oxford, UK: Clarendon Press.

Jelinek, F. (1976). Continuous speech recognition by statistical methods. *Proc. of the IEEE*, *64*(4), p.532–556.

Jelinek, F. (1980). Interpolated estimation of Markov source parameters from sparse data. In *Proc. of the Workshop on Pattern Recognition in Practice*. Amsterdam, The Netherlands.

Jelinek, F. (1991). Up from trigrams! - the struggle for improved language models. In *Proc. of the Eurospeech*. Genoa, Italy.

Jelinek, F. (2009). The dawn of statistical ASR and MT. *Computational Linguistics*, *35*(4), p.483–494.

Jelinek, F., Mercer, R. L., Bahl, L. R. & Baker, J. K. (1977). Perplexity—a measure of the difficulty of speech recognition tasks. *The Journal of the Acoustical Society of America*, *62*(S1), p.S63–S63.

Johnson, W. E. (1932). Probability: The deductive and inductive problems. *Mind*, *41*(164), p.409–423.

Kaplan, B. & Maxwell, J. A. (2005). Qualitative research methods for evaluating computer information systems. In J. G. Anderson & C. E. Aydin (Eds.), *Evaluating the organizational impact of healthcare information systems* (2nd Ed., p.30–55). New York, USA: Springer.

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

Kilgarriff, A. & Grefenstette, G. (2003). Introduction to the special issue on the web as corpus. *Computational Linguistics*, *29*(3), p.333–347.

Klakow, D. (1998). Log-linear interpolation of language models. In *Proc. of the ICSLP*. Sydney, Australia.

Kneser, R. & Ney, H. (1995). Improved backing-off for m-gram language modeling. In *Proc. of the ICASSP* (Vol. 1). Detroit, USA.

Kobayashi, A., Onoe, K., Imai, T. & Ando, A. (1998). Time dependent language model for broadcast news transcription and its post-correction. In *Proc. of the ICSLP*. Sydney, Australia.

Kuhn, R. & Mori, R. D. (1990). A cache-based natural language model for speech recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, *12*(6), p.570–583.

Lange, P. (2014a). *A review of speech and text corpora in the medical domain*. School of Computing, Staffordshire University. Discipline Specific Module, Master by Research.

Lange, P. (2014b). *Generating a Language Model Performance Baseline for ASR in the Medical Domain*. School of Computing, Staffordshire University. Advanced Research Module, Master by Research.

Lau, R., Rosenfeld, R. & Roukos, S. (1993). Trigger-based language models: a maximum entropy approach. In *Proc. of the ICASSP* (Vol. 2). Minneapolis, USA.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *10*(8), p.707–710.

Lidstone, G. J. (1920). Note on the general case of the Bayes-Laplace formula for inductive or a posteriori probabilities. *Transactions of the Faculty of Actuaries*, *8*(p.182-192).

Martin, S. C., Liermann, J. & Ney, H. (1997). Adaptive topic-dependent language modelling using word-based varigrams. In *Proc. Eurospeech*. Rhodes, Greece.

Martin, S., Liermann, J. & Ney, H. (1998). Algorithms for bigram and trigram word clustering. *Speech Communication*, *24*(1), p.19–37.

Mikolov, T. (2012). *Statistical Language Models Based on Neural Networks* (Doctoral dissertation, Faculty of Information Technology, Brno University of Technology, Brno, Czech Republic).

Mikolov, T., Karafiát, M., Burget, L. et. al. (2010). Recurrent neural network based language model. In *Proc. of the Interspeech*. Makuhari, Japan.

Moore, R. C. & Lewis, W. (2010). Intelligent selection of language model training data. In *Proc. of the ACL*. Uppsala, Sweden.

Ney, H. & Essen, U. (1991). On smoothing techniques for bigram-based natural language modelling. In *Proc. of the ICASSP*. Toronto, Canada.

Ney, H., Essen, U. & Kneser, R. (1994). On structuring probabilistic dependences in stochastic language modelling. *Computer Speech & Language*, *8*(1), p.1–38.

Oku, T., Fujita, Y., Kobayashi, A. & Sato, S. (2013). Progressive language model adaptation for disaster broadcasting with closed-captions. In *Proc. of the APSIPA*. Kaohsiung, Taiwan.

Pellegrini, T. & Lamel, L. (2008). Are audio or textual training data more important for ASR in less-represented languages? In *Proc. of the SLTU*. Hanoi, Vietnam.

Pereira, F. (2000). Formal grammar and information theory: together again? *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, *358*(1769), p.1239–1253.

Povey, D., Ghoshal, A., Boulianne, G. et. al. (2011). The Kaldi Speech Recognition Toolkit. In *Proc. of the ASRU*. Hilton Waikoloa Village, USA.

Rosenfeld, R. (1995). Optimizing lexical and N-gram coverage via judicious use of linguistic data. In *Proc. of the Eurospeech*. Madrid, Spain.

Rosenfeld, R. (2005). *Adaptive statistical language modeling: A maximum entropy approach* (Doctoral dissertation, School of Computer Science, Carnegie Mellon University, Pittsburgh, USA).

Schäfer, R., Barbaresi, A. & Bildhauer, F. (2013). The good, the bad, and the hazy: Design decisions in web corpus construction. In *Proc. of the WAC*. Lancaster, UK.

Schlippe, T., Gren, L., Vu, N. T. & Schultz, T. (2013). Unsupervised language model adaptation for automatic speech recognition of broadcast news using web 2.0. In *Proc. of the Interspeech*. Lyon, France.

Schwenk, H. (2007). Continuous space language models. *Computer Speech & Language*, *21*(3), p.492–518.

Schwenk, H. & Gauvain, J.-L. (2003). Using continuous space language models for conversational speech recognition. In *Proc. of the SSPR*. Tokyo, Japan.

Simons, M., Ney, H. & Martin, S. C. (1997). Distant bigram language modelling using maximum entropy. In *Proc. of the ICASSP* (Vol. 2). Munich, Germany.

Siu, M. & Ostendorf, M. (2000). Variable n-grams and extensions for conversational speech language modeling. *IEEE Trans. on Speech and Audio Processing*, *8*(1), p.63–75.

Stolcke, A., Konig, Y. & Weintraub, M. (1997). Explicit word error minimization in n-best list rescoring. In *Proc. of the Eurospeech*. Rhodes, Greece.

Su, Y., Jelinek, F. & Khudanpur, S. (2007). Large-scale random forest language models for speech recognition. In *Proc. of the Interspeech*. Antwerp, Belgium.

Suchomel, V. & Pomikálek, J. (2012). Efficient web crawling for large text corpora. In *Proc. of the WAC*. Lyon, France.

Versley, Y. & Panchenko, Y. (2012). Not just bigger: Towards better-quality Web corpora. In *Proc. of the WAC*. Lyon, France.

Wahlster, W. (2000). *Verbmobil: foundations of speech-to-speech translation*. Berlin, Germany: Springer Science & Business Media.

Ward, W. & Issar, S. (1996). A class based language model for speech recognition. In *Proc. of the ICASSP*. Atlanta, USA.

Whittaker, E. W. D. (2000). *Statistical language modelling for automatic speech recognition of Russian and English* (Doctoral dissertation, University of Cambridge, Cambridge, UK).

Whittaker, E. W. D. & Woodland, P. C. (2001). Efficient class-based language modelling for very large vocabularies. In *Proc. of the ICASSP* (Vol. 1). Salt Lake City, USA.

Wu, J. & Khudanpur, S. (2000). Efficient training methods for maximum entropy language modeling. In *Proc. of the Interspeech*. Beijing, China.

Xu, P. & Jelinek, F. (2004). Random forests in language modeling. In *Proc. of EMNLP* (Vol. 4). Barcelona, Spain.

Young, S. (2008). HMMs and related speech recognition technologies. In J. Benesty, M. M. Sondhi & Y. A. Huang (Eds.), *Springer Handbook of Speech Processing* (p.539–558). Berlin, Germany: Springer.

Zhang, X., Trmal, J., Povey, D. & Khudanpur, S. (2014). Improving deep neural network acoustic models using generalized maxout networks. In *Proc. of the ICASSP*. Florence, Italy.

Zhou, G. & Lua, K. T. (1998). Word association and MI-Trigger-based language modeling. In *Proc. of the ICCL* (Vol. 2). Montreal, Canada.

Acronyms

ASR	Automatic Speech Recognition
FFNN	Feedforward Neural Network
fMLLR	Feature Space Maximum Likelihood Linear Regression
KN	Kneser-Ney
LDA	Linear Discriminant Analysis
MFCC	Mel Frequency Cepstral Coefficients
mKN	modified Kneser-Ney
MLLT	Maximum Likelihood Linear Transform
NLP	Natural Language Processing
OOV	out of vocabulary
RNN	Recurrent Neural Network
WER	word error rate

Glossary

Context	The context is the words in a sequence used to estimate the next word in the sequence by a language model
Domain	The domain of a text is its topic of discourse
History	The history is the words in a sequence used to estimate the next word in the sequence by a language model
In-domain	Training data is considered as in-domain when it is from the same domain as the use case
Token	A token is an appearance of a type in a text
Type	Types are the different words that can occur in a text
Utterance	An utterance is a vocal expression