BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2020
http://excel.fit.vutbr.cz

Best possible speech recognizer on your own data

Tomáš Sýkora*

Abstract
Many state-of-the-art results in different machine learning areas are presented on day-to-day basis. By adjusting these systems to perform perfectly on a specific subset of the general data, huge improvements may be achieved in their resulting accuracy. Usage of domain adaptation in automatic speech recognition can bring us to production level models capable of transcribing difficult and noisy customer conversations way more accurately than the general models trained on all kinds of language and speech data. In this work I present 12.7% word error rate improvement in our speech recognition task over the general domain speech recognizer from Google. The improvement was achieved by both very precise annotation and preparation of domain data and by combining state-of-the-art architectures and algorithms. The described system was successfully integrated into a production environment of the Parrot transcription company founded by, among other team members, current and former faculty students, which drastically increased performance of the human transcribers.

Keywords: automatic speech recognition — domain adaptation — conversational speech

Supplementary Material: *N/A*

*xsykor25@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

Automatic speech recognition (ASR) has achieved many improvements over the last decade, similarly to other machine learning areas. New algorithms and more computational power allowed researchers to feed the systems with much more data than before, which markedly boosted results and accuracies of such systems. As a result, machine learning products can be now applied in various day-to-day scenarios where they automate the previously used manual processes and save both time and money of companies using them.

Automatic speech recognition is a system capable of transcribing speech from audio signal form to text. In Parrot, a court reporting transcription company, we integrate automatic speech recognition as a helper tool for our human transcribers. After obtaining a transcription request from a customer and generating the automatic transcription, the human transcribers fix errors in the ASR output using a specialised editor tool which is way more efficient than transcribing the whole recordings from scratch manually. To improve experience and efficiency of the human transcribers even more, we decided to build a more accurate automatic speech recognition system. As until now we've been using a general speech recognition, I show in this paper that a domain specific system can outperform the general one in terms of word error rate (WER).

The main reason for this being possible is no need of transcribing all kinds of target domains, acoustic setups and specific language dialects. Instead, we focus only on scenarios that occur commonly in our use case. Although this way our model performs worse in general, it performs way better on our target data where it knows the data well as it was trained on similar ones.

The goal of this work was to achieve higher transcription accuracy on our legal based audio conversations than the one obtained with the general domain model. This was accomplished by very precise data selection, annotation and preparation followed by the state-of-the-art language and acoustic models. Precise manual transcriptions cleaning, robust text normalization of unlabeled data, various audio augmentations, speaker adaptations, recurrent neural network language models and time-delayed neural architectures resulted in an ASR system with more than 12.7% improvement in WER in comparison to a general ASR model developed by Google[1].

2. Automatic speech recognition systems

To solve the large vocabulary continuous speech recognition (LVCSR) [1] problem, ASR systems stand on complex architectures composed of several subsystems. Specifically, acoustic and languages models combined together in a decoding network as depicted in Figure 1.

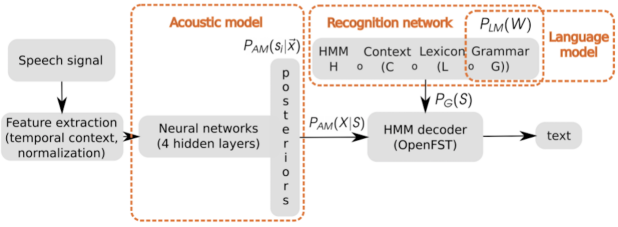

Figure 1. *Components of an automatic speech recognition system transcribing a speech signal to text (figure obtained from [2]).*

Acoustic models (AM) are trained to match every word in a speech signal to a sequence of acoustic units called phonemes. To do this, Hidden Markov models (HMM) [3] are one of the state-of-the-art approaches to solve this sub-problem. To simulate the human perception of speech where context plays an important role, language models (LM) predict probabilities of word sequences in a given language. The acoustic and language model combination can be expressed in a form of the Bayesian rule shown in Equation 1, where $P(X|W)$ is modeled by an acoustic model (probability of the acoustic observation X given the word sequence W), while a language model estimates and models the second part of the rule $P(W)$ (apriori probability of a word sequence W). Assuming $P(X)$ is the same for all possible transcriptions, we do not have to consider it and only the numerator part of the rule remains to be realised by an ASR system.

$$W^* = \arg\max_w P(W|X) = argmax_w \frac{P(X|W)P(W)}{P(X)} \quad (1)$$

Acoustic models

Current state-of-the-art approaches in HMM states representation use deep neural networks (DNN) [4] which replaced the previously used Gaussian Mixture Models (GMMs). However, GMMs are still found useful as a preprocessing step before the neural net training. As mentioned in [5], they help to force align phonemes in the individual speech utterances before sending them into an actual neural network based model.

The accuracy of a DNN during the training is approximated by an objective function. A basic objective function commonly used for a 1-of-K classification problem is the multi-class cross-entropy (CE). However, later studies proved that replacing the cross-entropy objective function with sequence-discriminative criteria (e.g. maximum mutual information (MMI)) improves the overall results in the automatic speech recognition task as it is a sequence classification problem [6]. Further on, usage of lattice-free version of the MMI objective function (LF-MMI) decreases the computational costs by omitting the need of precomputation of the lattices for all possible word sequences (only the reference word sequence lattices remain) and avoiding the initial training with a CE model to create a precise weights initialisation [5].

Due to dynamic nature of speech ASR systems have to model temporal relationships between acoustic events, while at the same time providing for invariance under translation in time. To fight this, the time-delay neural network (TDNN) architecture [7] takes a window of both past and future feature frames to recognise a single phoneme. Additionally, as shown in Figure 2, later studies showed that not all connections between all frames are necessary, which significantly improves performance and lowers the amount of computations needed during training [8].

To further improve the prediction and generalization abilities of acoustic models, different speaker adaptations may be applied. i-vectors, low-dimensional features capable of characterizing speakers, are often

[1] https://cloud.google.com/speech-to-text/

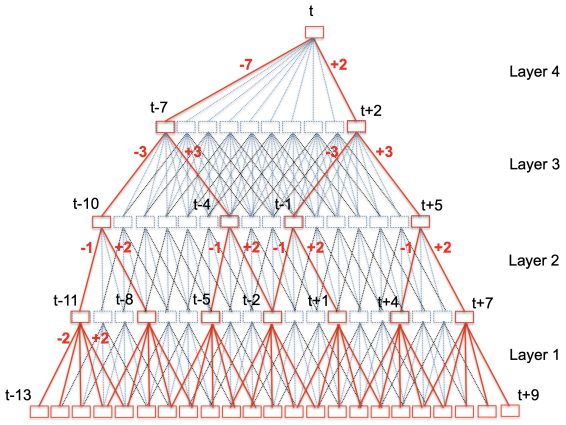

Figure 2. *Pruned time-delay neural network as presented in [8] computes with sub-sampling (red) instead of using all connections (blue+red).*

taken as additional input to an acoustic model besides the main (*mfcc*) features [9].

Language models

Language models (LM) are applied to estimate word sequences apriori probabilities in a given spoken language (e.g. English). The most widely used statistical language model is the *n*-gram LM. Its goal is to identify occurrence counts of sets of *n*, *n*-1, ..., 1 words in a large text corpus. The *n*-gram model is then used to predict *n*-th word given a *n*-1 long word sequence. Additionally, techniques like Kneser-Ney smoothing [10] were introduced to improve the original *n*-gram algorithm. If multiple different domain text corpuses are available, building separate domain-specific language models for each corpus individually and applying an interpolation technique to combine them into a single language model optimised for the target use case showed better performance in comparison to training a single LM on a big mixed-text corpus [11].

To further increase the language modeling capacities, recurrent neural network (RNN) models come in place. Although RNN language models (RNNLMs) show better results in comparison to the *n*-gram models, because of the RNNLM theoretically infinite history input lengths, it is technically impossible to compile them into a static decoding graph. Thus RNNLMs are usually not directly used in the decoding. Instead, lattice rescoring is a common approach to take the advantage of the recurrent models in the decoding process [12]. After a word lattice is generated from the 1st-pass decoding, it is then rescored with an RNNLM.

As a conjunction of acoustic and language modelling, modelling inter-word silence probabilities proved to consistently improve the overall accuracy of ASR systems [13]. As an example, zero-silence is less likely to follow the word *White* in "Gandalf the White said," than in "The White House said.".

Kaldi - speech recognition toolkit

Kaldi[2], introduced in [14], is a free open-source project consisting of many utilities for different parts of ASR pipelines from feature extraction, through GMM/DNN based HHMs training and speaker adaptation techniques, to decoding graphs and lattice rescoring algorithms implementations. Besides the individual components, algorithms and utilities, Kaldi also provides multiple prearranged pipelines called recipes containing state-of-the-art setups on different generally known speech datasets. Kaldi toolkit utilities were mostly used in this project.

3. Conversational speech dataset preparation

The data used to train both language and acoustic models had to perfectly represent the target domain. The accuracy of the data annotation had to meet high expectations. The whole training dataset for the acoustic model was composed of two subsets - the real data from the company customers (29 hours) and a dataset of the U.S. Supreme Court public hearings transcriptions[3] (the whole dataset contains 7000 hours but only a 500h subset was used in this project so far). Our target domain consisted of different audio quality types:

- Formal depositions - indoor environment, quiet room, mostly clear speech, high-quality microphone based on a table in front of the speakers.
- Witness statements - indoor environment, less formal environment, more spontaneous and sentimental discussion, middle quality microphones.
- Different less frequently represented scenarios - prison phone calls, police bodycams, street conversations etc.

Manual annotation

Acoustic data annotation was performed in two steps. First, the whole recordings (5-400 minutes long) were transcribed in a way that suits the customer - with numerical characters, abbreviations, punctuation etc. Correct speaker labels were assigned to individual speech parts which segmented the audio into smaller speaker segments. In the second step, the speaker segments were automatically assigned with start/end timestamps using the *gentle*[4] force alignment tool. The timestamps

[2] https://kaldi-asr.org/

[3] https://www.supremecourt.gov/oral_arguments

[4] https://github.com/lowerquality/gentle

were thereafter manually corrected by human annotators in a proprietary designed annotation tool. Besides that, the human annotators were instructed to normalize the original transcriptions to a fully verbatim form (e.g. year 1950 replaced with 19 50 if pronounced as "nineteen fifty" etc.).

Language model text corpuses

To train the language models (both *n*-gram and RNNLM), different text corpuses were experimented with. Firstly, the company transcriptions were used as one text corpus (17k sentences). Secondly, the whole Supreme Court hearings dataset transcriptions created the second text corpus (4M sentences). The rest consisted of general language text corpus (42M sentences subset of Google Billion Word Benchmark [15]), a set of legal deposition transcriptions collected on the Internet (3M of conversational data) and a set of publicly available court decisions[5] (20M sentences of non-conversational data).

Automatic data preprocessing

As only the company transcriptions were manually normalized and transformed to the fully verbatim form, automatic normalization had to take place for all texts available. The normalization replaced all numerical values and special characters like dollars and percents with their alphabetical equivalents in an appropriate way (*'70s* as *seventies*, *$100,000* as *one hundred thousand dollars* etc.). The letter case was processed only at the sentence beginnings, the rest of words remained with the original case; each word at a sentence beginning was decided to remain capital or was lowercased with help of a pretrained named entity recognition tool *spacy*[6]. All punctuation was removed except in special cases like *hm-mm* or *uh-huh* and single quotes (*don't*). Abbreviations were split into single letters and several other minor preprocessing steps were performed.

Input text:

MMy name Bond, james bond. A CIA agent. Ian Flaming introduced me in 1953 in a Casino Royale novel, later filmed in 2006 for $10 million.

Text after manual correction and partial normalization:

My name is Bond, James Bond. A CIA agent. Ian Flaming introduced me in 19 53 in a Casino Royale novel, later filmed in 2006 for $10 million.

Text after automatic normalization:

my name is Bond James Bond a C I A agent Ian Flaming introduced me in nineteen fifty three in a Casino Royale novel later filmed in two thousand six for ten million dollars

Due to the LF-MMI objective function sensitivity to incorrect transcripts, the Supreme Court acoustic data were cleaned from the utterances with supposedly incorrect transcriptions. The cleaning was performed using a *kaldi* script[7]. The script removed utterances which, after decoding with a biased LM (trained on the transcriptions themselves), the lattice oracle path was still far from the transcript. This way, 12% of the acoustic data was considered to be incorrect and thus removed from the dataset. The removed utterances often contained crosstalks or the audio was cut in the middle of the utterance text.

Data augmentation

To increase the size of the company dataset and for the Supreme Court data to better match the target acoustic setup, all of the data underwent an augmentation procedure using speed perturbation and the room impulse responses dataset described in [16] (RIR noises). In the paper, 3-fold speed perturbation was used in all experiments. Additionally, improvements of both 2 and 3-fold augmentations using RIR noises in addition to speed perturbation were presented. As our company dataset size was too small at the moment of training, 3-fold speed perturbation was applied to create different speed versions of the original data. Thereafter, the amount of augmented data with RIR noises was increased from 3 to 4-fold in comparison to the original paper. Supreme Court dataset was doubled using RIR noises. No speed perturbation took place as the amount of the Supreme Court data was sufficient. All mentioned augmentation procedures were performed using appropriate *kaldi* scripts.

Additionally, all utterances were concatenated with randomly selected 30ms silence audio chunks (30ms to both start and end of an utterance) to prevent HMM first and last silence representing states from assigning speech phonemes to them, which would be the case for utterances where there were no silence frames at their beginnings and/or ends.

[5] https://case.law/bulk

[6] https://spacy.io/

[7] https://github.com/kaldi-asr/kaldi/blob/master/egs/wsj/s5/steps/cleanup/clean_and_segment_data_nnet3.sh

4. Experiments

Throughout the experiments, the company dataset was split into an 11.5 hour test set and an 18 hour train set, both with unique speakers. 500 hours were separated from the U.S. Supreme Court dataset to increase the train set size and another 60 hours Supreme Court subset was created as a second test set. Both company and Supreme Court sets were augmented. The company 18 hours were augmented to 220 hours using 4-fold RIR noises and 3-fold speed perturbation. The Supreme Court data were augmented from 500 hours to 990 hours using 2-fold RIR noises augmentation. The separate improvements are described below.

Language models

Several LM experiments were performed to prove that the interpolation of several narrow domain *n*-gram LMs can outperform one large *n*-gram LM. First, four different-domain 4-gram language models were trained on the text corpuses described in Table 1. An open-source tool KenLM[8] was used to train the individual LMs and the final interpolated LM.

Table 1. *4-gram language models tested on a subset of 9k sentences of the company transcriptions not seen during the training. Log-linear weights were assigned by the interpolation procedure using the KenLM tool.*

Dataset	Size	Weight	Perplexity
General	42M	0.2	510
Legal general	23M	0.1	427
Supreme Court	4M	0.2	407
Parrot transcriptions	17k	0.5	146
Interpolated LM	-	-	126

In another experiment, the same text corpuses were used differently. The out-of-domain corpus (Google Billion Words) was not used at all. The Supreme Court dataset was split into two subsets, transcriptions before and after the year 2000. Finally, the legal general corpus was split into two subset, dataset of Court Decisions (non-conversational texts) and legal speech transcriptions collected on the Internet (conversational texts). Another difference in comparison to the previous LM experiment was improved text normalization capable of correct handling of more specific use cases than the previous normalization version. To sustain the comparability of the test set, the original normalization was applied on it in the second experiment.

Two different RNNLMs for lattice rescoring were trained on the same data splits used by the *n*-gram LMs. Their contributions are shown the tables 3 and 4.

[8] https://kheafield.com/code/kenlm/

Table 2. *4-gram language models tested on a subset of 9k sentences of company transcriptions not seen during the training. Log-linear weights were assigned by the interpolation procedure. The text corpuses from in Table 1 were differently divided and more robust normalization was applied on them. The robust normalization removed a big part of the Court decisions corpus.*

Dataset	Size	Weight	Perplexity
Court decisions	15.5M	0.2	340
Legal transcriptions	2.5M	0.55	176
Supreme Court	1.5M	-0.1	337
Supreme Court from before 2000	2.5M	-0.1	340
Parrot transcriptions	17k	0.45	139
Interpolated LM	-	-	123

Acoustic models

The acoustic model architecture was based on the kaldi Switchboard recipe[9]. Firstly, a sequence of GMM based models was trained, each of which used phoneme level alignments from the previously trained GMM model. Starting with a monophone training on a subset of the shortest utterances (to minimize incorrect phoneme alignments), continuing with bigger models trained on bigger subsets, ending with a triphone GMM model with speaker adaptation, which thereafter aligned the utterance phonemes before being passed into the TDNN model.

Table 3. *Convolutional time-delay (cnn-tdnnf) models trained on the cleaned and augmented Supreme Court dataset (s.c., 500h augmented to 990h) and 18 hours of our company data. The 18 hours of Parrot data were extended by augmentations to 220h. test1 is an 11.5 hour test set of the company data, test2 is a 60 hour subset of the Supreme Court dataset, both not seen during the training.*

	Training info:		WER [%]	
AM	LM	Dataset	test1	test2
google	unknown	unknown	35.0	-
cnn-tdnnf	4-gram	990h s.c.	38.3	13.8
cnn-tdnnf	4-gram + lat. rescoring	990h s.c.	34.9	15.1
cnn-tdnnf	4-gram	18h parrot + 200h aug. + 990h s.c.	27.63	13.96
cnn-tdnnf	4-gram + lat. rescoring	18h parrot + 200h aug. + 990h s.c.	**24.21**	-

[9] https://github.com/kaldi-asr/kaldi/tree/master/egs/swbd

Experiments with different data setups were run which is demonstrated in the tables 3 and 4.

Table 4. *Improved 4-gram language model from Table 2 and a new RNNLM trained on the newly normalized corpuses from the same table used with the same AM as in the Table 4. test1 is the company 11.5h test set same as the test1 from Table 3. test3 is the same set except with the improved text normalization.*

Training info:			WER [%]	
AM	**LM**	**Dataset**	test1	test3
google	unknown	unknown	35.0	35.1
cnn-tdnnf	4-gram (v2)	18h parrot + 200h aug. + 990h s.c.	24.65	24.42
cnn-tdnnf	4-gram (v2) + lat. rescoring	18h parrot + 200h aug. + 990h s.c.	22.59	**22.32**

The best WER was achieved by decoding with the beam size 15, lattice pruning beam 12 and rescoring lattice beam 12. Higher values resulted in around 0.01% improvement but drastically increased the computing time thus the lower beam values seem to be the most reasonable.

5. Conclusions

A combination of proven state-of-the-art automatic speech recognition components was presented in this work. The paper demonstrates how to prepare conversational speech data from a specific domain to achieve higher accuracy than a generic ASR model. With only 18 hours of the target domain data in combination with a public similar domain dataset of 500 audio hours, a 12.7% WER improvement was gained in comparison to a general ASR model from Google. The paper shows how the individual components and data preparation steps help with increasing the resulting accuracy of the system.

Improvements of several of the mentioned components weren't measured separately thus it's not clear how big contribution they brought. Such experiments have to be run in the future. Also, no time was spent to tune the hyper-parameters of the individual components which can become an object of the future research. Additionally, a bigger target domain dataset could boost the final results even more.

Acknowledgements

I would like to thank my supervisor Ing. Igor Szőke Ph.D. for his help and advice in this project.

References

[1] Steve Young. A review of large-vocabulary continuous-speech. *IEEE Signal Processing Magazine*, 13(5):45–, Sep. 1996.

[2] Karel Veselý. The project kaldi – open source speech recognition. online, 2018.

[3] Mark Stamp. A revealing introduction to hidden markov models, 2004.

[4] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, Nov 2012.

[5] Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahremani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur. Purely sequence-trained neural networks for asr based on lattice-free mmi. pages 2751–2755, 09 2016.

[6] Karel Veselý, Arnab Ghoshal, Lukáš Burget, and Daniel Povey. Sequence-discriminative training of deep neural networks. In *Proceedings of Interspeech 2013*, number 8, pages 2345–2349. International Speech Communication Association, 2013.

[7] Alex Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. Phoneme recognition using time-delay neural networks. *IEEE transactions on acoustics, speech, and signal processing*, 37(3):328–339, 1989.

[8] Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. A time delay neural network architecture for efficient modeling of long temporal contexts. In *INTERSPEECH*, 2015.

[9] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny. Speaker adaptation of neural network acoustic models using i-vectors. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 55–59, Dec 2013.

[10] Reinhard Kneser and Hermann Ney. Improved backing-off for m-gram language modeling. *1995 International Conference on Acoustics, Speech, and Signal Processing*, 1:181–184 vol.1, 1995.

[11] Ernest Pusateri, Christophe Van Gysel, Rami Botros, Sameer Badaskar, Mirko Hannemann, Youssef Oualil, and Ilya Oparin. Connecting and

comparing language model interpolation techniques. *arXiv preprint arXiv:1908.09738*, 2019.

[12] Hainan Xu, Tongfei Chen, Dongji Gao, Yiming Wang, Ke Li, Nagendra Goel, Yishay Carmiel, Daniel Povey, and Sanjeev Khudanpur. A pruned rnnlm lattice-rescoring algorithm for automatic speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5929–5933. IEEE, 2018.

[13] Guoguo Chen, Hainan Xu, Minhua Wu, Daniel Povey, and Sanjeev Khudanpur. Pronunciation and silence probability modeling for asr. In *INTERSPEECH*, 2015.

[14] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukáš Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlíček, Yanmin Qian, Petr Schwarz, Jan Silovský, Georg Stemmer, and Karel Vesel. The kaldi speech recognition toolkit. *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*, 01 2011.

[15] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling, 2013.

[16] Tom Ko, Vijayaditya Peddinti, Daniel Povey, Michael L Seltzer, and Sanjeev Khudanpur. A study on data augmentation of reverberant speech for robust speech recognition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5220–5224. IEEE, 2017.