Computational Linguistics in the Netherlands Journal 11 (2021) 27-34 Submitted 10/2021; Published 12/2021

A Hybrid ASR System for Southern Dutch

Bob Van Dyck* BOB.VANDYCK@KULEUVEN.BE
Bagher BabaAli** BABAALI@UT.AC.IR
Dirk Van Comperolle* DIRK.VANCOMPERNOLLE@KULEUVEN.BE

*Department of ESAT - KU Leuven, Leuven, Belgium

**School of Mathematics, Statistics and Computer Science - University of Tehran, Teheran, Iran

Abstract

Classical hybrid models for automatic speech recognition were recently outperformed by end-to-end models on popular benchmarks such as LibriSpeech. However, in many real life situations, hybrid systems can prevail due to independent training, optimization and tuning of the acoustic and language models. In this work, we implemented a state-of-the-art hybrid system for Southern Dutch. For the acoustic model, we train a HMM-DNN on 155 hrs of the Corpus Gesproken Nederlands (CGN) with a rather standard Kaldi recipe. As reference, we reused language models developed during our N-Best 2008 evaluation. We further investigated the effect of language model order and size on WER for a variety of test sets (held out data from CGN, N-Best dev and test sets). Best results, 10.12% WER on the N-Best test set, are obtained with a 400k lexicon and a 4-gram language model (with 231M parameters). This new hybrid system outperforms our older HMM-GMM based N-Best system by over 40%. Pruning away 90% of the LM parameters yields a compact model suitable for small scale real-time apps while only taking a 10% relative hit on performance.

1. Introduction

Hybrid models for automatic speech recognition (ASR) were recently outperformed by end-to-end (E2E) models on popular benchmarks such as LibriSpeech. However, in many real life situations, hybrid systems can prevail due to independent training, optimization and tuning of their components. The classical hybrid ASR system is comprised of an acoustic model (AM) and a language model (LM). The AM uses a Deep Neural Network (DNN) for estimating the observation probabilities (for each phoneme or phonetic state) of the Hidden Markov Model (HMM). While these components require separate training and don't allow for joint optimization with a downstream objective, they do allow for independent training, optimization and tuning. In many real life situations, the advantages of independent training of the separate components (as in the hybrid setup) may outweigh the advantage of having a global optimization objective (as in end-to-end systems). This is generally the case with limited resources (a few hundred or fewer hours of data) or with significant language model mismatch between train and test data, although self-supervised pre-training methods are alleviating the need for labelled data. Current demands for our speech recognizer are exemplary of this need as it often includes domain specific tasks such as lecture transcriptions, meeting transcriptions, etc.

In this paper we present our hybrid ASR model for Southern Dutch, implemented in Kaldi (Povey et al. 2011). The aim is to create an acoustic model for Southern Dutch that is broadly applicable and a global system that is adaptable to different use-cases through lexicon and language model adaptations. To this end, we investigate the effect of language model order and size on word error rate for held out data from the Spoken Dutch Corpus and the N-Best 2008 benchmark.

2. Related works

Extensive results on the N-Best 2008 benchmark were obtained during the N-Best 2008 evaluation campaign by van Leeuwen et al. (2009) and publications such as the ESAT 2008 system (Demuynck et al. 2009) and SHoUT (Huijbregts et al. 2009). All these results were obtained with "classical" HMM-GMM systems in which the observation probabilities of the HMM are computed via Gaussian Mixture Models (GMM). Since then GMMs have been phased out in favor of DNNs. This has already been the case for ESAT systems as used in the STON project (Verwimp et al. 2016) or for the Code Switching system presented in Yılmaz et al. (2018). However, these systems used rather simple multilayer perceptrons for the DNN which have also been replaced since by more complex architectures. Neither did they present benchmark results on N-Best. In this paper we present a state-of-the-art hybrid DNN and benchmark results on N-Best. Only a limited number of results have been published so far with Dutch end-to-end systems. However, performance as pulished by Röpke et al. (2019) is by no means competitive with the hybrid approach presented here.

3. Data

3.1 Speech corpora

3.1.1 Spoken Dutch Corpus

The Spoken Dutch Corpus (*Corpus Gesproken Nederlands, CGN*) (Oostdijk 2000) is a manually orthographically annotated speech corpus of around 900 hours of contemporary Dutch, of which 270 hours correspond to Southern Dutch. We only include Southern Dutch and exclude all narrowband telephone speech and spontaneous conversational speech, which correspond respectively to components C,D and component A of CGN. The resulting 155 hours of audio is randomly partitioned into a training (90%) and development (10%) set, respectively called *CGN-train* and *CGN-dev*.

3.1.2 N-Best 2008

N-Best 2008 is a Dutch benchmark for Large Vocabulary Speech Recognition (Kessens and van Leeuwen 2007). We evaluate only on Southern Dutch broadcast news and use the corresponding evaluation (2h) and development materials (1h), further called *Nbest-test* and *Nbest-dev*. Note the development materials of N-best were taken from CGN and *Nbest-dev* is thus included in the training data.

3.2 Language and pronunciation model

As reference language model, we use the ESAT 2008 N-best n-gram language model (Demuynck et al. 2009) without further adaptations. The training material for the LM was obtained from two resources: the Dutch publisher PCM (360 million words) and the Flemish Mediargus (1,436 million words) (Kessens and van Leeuwen 2007). The pronunciation model or lexicon is based on Fonilex (Mertens and Vercammen 1997), which provides multiple Southern Dutch phonemic transcriptions for 170k common Dutch words. For missing words, the lexicon is supplemented with automatically generated phonetic transcriptions based on rule-based inflection, acronym handling and a grapheme-to-phoneme module (Demuynck et al. 2009). Two version of the lexicon are considered, a 400k and 100k version, each with a corresponding LM. The words in both lexicons are chosen based on word frequency in the LM training material. Their lexical coverage over the evaluation sets expressed as out-of-vocabulary (OOV) rate can be found in Table 1.

	OOV		
Lexicon	*CGN-dev*	*Nbest-dev*	*Nbest-test*
100k	5.46	2.27	2.72
400k	2.80	1.38	1.75

Table (1) OOV-rates (%) for lexicon sizes 100k and 400k

4. Hybrid ASR system

We consider a standard hidden Markov model (HMM) based hybrid ASR system, comprised of an acoustic model, pronunciation lexicon and n-gram language model. For the acoustic model, we train a time delayed neural network (TDNN) with the Lattice-free Maximal Mutual Information criterion (Povey et al. 2018) on *CGN-train*. A HMM-GMM system is used to compute the alignments and build a phonetic-context decision tree needed for training the neural network. The remaining components of the ASR system are represented in a decoding graph. For tuning some of the hyperparameters we used *CGN-dev*. The model implementation is done in Kaldi, using the NNET3 library for the TDNN, and is similar to the Switchboard recipe[1].

4.1 HMM-GMM system

The purpose of the HMM-GMM system is to build a phonetic-context decision tree and to compute alignments over the datasets. The HMM-GMM uses 13-dim mel frequency cepstral coefficents (MFCC) features extracted with a 25 ms window, a 10 ms shift, 23 mel frequency bins, DCT transformation and truncation to 13 coefficients. We train the HMM-GMM in several stages, the final stage being speaker-adaptive training of a triphone model on MFCC+LDA+MLLT features as in the standard Kaldi pipeline (Rath et al. 2013). This results in 3065 triphone states.

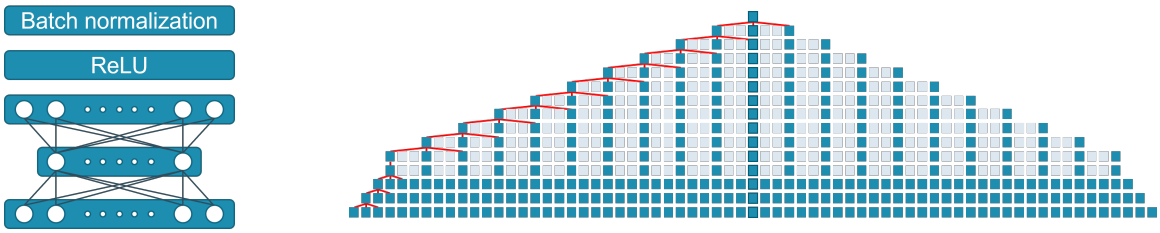
4.2 TDNN architecture

We train a sub-sampled time delayed neural network (TDNN) triphone model on 40-dim MFCCs and delta's. Again, MFCCs are extracted with a 25 ms window and a 10 ms shift, but now with 40 mel frequency bins and without cepstral truncation. We did not apply i-vector adaptation, since in our experience i-vectors are not effective in scenario's with short speaker fragments. We do apply cepstral mean normalization (CMN) on a per-speaker basis over a 6 s window. The TDNN has 14 TDNN-F layers (Povey et al. 2018) and each TDNN-F layer is 1536 dimensional, with a 160 dimensional low-rank weight factorization, ReLU activation and batch normalisation, as shown in Figure 1a. We splice frames at offset {-1, 0 1} for the first three TDNN-F layers, {0} for the next one and {-3, 0, 3} for the remaining, as shown in Figure 1b. Between the TDNN-F layers, skip connections occur with a probability of 0.66. Last layers are a 256-dim fully connected layer, followed by a 3065-dim output layer. The resulting TDNN has a 700 ms receptive field and 17M parameters. To reduce computation, we use a frame-subsampling factor of 3, meaning the network outputs are evaluated every third frame, both during training and inference.

4.3 Data augmentation

We augment the training data by applying speed and volume perturbations (Ko et al. 2015, Rath et al. 2013). Both augmentation methods result in mean shifts in the MFCC domain, similar to vocal tract length normalisation. Speed perturbation is achieved by resampling the audio with speed factors 0.9, 1.0 and 1.1, effectively tripling the training data. For volume perturbations, audio is scaled by a factor drawn from a uniform distribution $[\frac{1}{8}, 2]$.

1. https://github.com/kaldi-asr/kaldi/tree/master/egs/swbd/s5c



(a) (b)

Figure (1) Time delay neural network with (a) a single TDNN-F layer and (b) a depiction of the sub-sampling procedure with activate time steps at each layer colored dark blue.

4.4 Neural network training

The TDNN is trained with the Lattice-free Maximal Mutual Information (LF-MMI) criterion (Povey et al. 2018). MMI is a discriminative objective function which aims to maximize the probability of the reference transcription, while minimizing the probability of all other transcriptions. Because sequence level training tends to overfit (Povey et al. 2016), two regularization methods are used. Firstly, a separate output layer with a standard cross-entropy objective is introduced. The cross-entropy objective is scaled by a constant factor 0.1 to compensate for it's larger dynamic range compared to the LF-MMI objective. Note the separate output layer is not used during inference. Secondly, we use a leaky HMM, which allows transition probabilities from each state in the HMM to every other state, with a *leaky-hmm-factor* of 0.1. Dropout is varied during training: none occurs for the first 20% of training, the dropout fraction increases linearly to 50% at 50% of training and then linearly drops to zero again by the end of training.

4.5 Decoding graph

The decoding graph can be interpreted as a composed weighted finite state transducer (WFST) of cascade H $\circ$ C $\circ$ L $\circ$ G. These components represent the HMM definition's (H), context-dependent phones (C), a pronunciation lexicon (L) and a language model (G). During decoding, a search graph is constructed for each utterance, where each path through the graph corresponds to a different hypothesis. Due to the sub-sampling of the AM outputs, the effective frame shift during decoding is 30 ms. Beam pruning is used to keep the search graph (or lattice) tractable. We use a beamwidth of 15 and keep the number of active states minimally 200 and maximally 7000. Upon saving the lattice, an additional pruning is performed to limit the number of hypotheses per time step to 8.

4.6 Evaluation

We report Word Error Rates based on the Levenshtein distance after normalizing both the reference transcription and hypothesis. Normalization consists of number substitutions according to writing conventions, decapitalization, and decompounding on hyphens. Compounding errors are ignored, as are non-speech sounds when the recognizer identified them as such. We report single-pass decoding results for the three evaluation datasets, but ultimately compare systems based on *Nbest-test*, since *CGN-dev* was used for designing the TDNN and *Nbest-dev* is contained in the training data.

5. Language model optimization

In real-world systems, optimizing the language model with respect to practical constraints is necessary. In a real-time system one preferably performs single-pass decoding to prevent latency issues due to rescoring. Since larger LMs are able to generate a more accurate lattice (provided a good

match between train and test data), the LM size is a limiting factor on the performance due to memory constraints. Moreover, memory constraints can also prohibit the compilation of the decoding graph HCLG, especially because the compilation cannot be parallelized. In our experience, 400 GB RAM is insufficient to successfully compile a decoding graph with a 400k 5-gram LM. In both scenario's, we want to maximize the language model size within the memory constraints. To this end, we investigate how we best reduce the LM size by pruning.

5.1 Pruning methods

We consider two pruning methods: reducing the order and entropy-based pruning. Entropy-based pruning aims to minimize the 'distance' between the distribution embodied by the original model and that of the pruned model (Stolcke 2000). This is achieved by scoring each n-gram according to the relative perplexity (or entropy) increase when it is removed and pruning all n-grams with an increase below a threshold θ. Note that back-off weights are recomputed after pruning. All adaptations to the reference language model are made using the SRILM toolkit (Stolcke 2002).

Entropy-based pruning removes n-grams that contribute less in terms of relative perplexity first, thus higher order n-grams will naturally be pruned first as they have low probabilities. Note that aggressive pruning, with a pruning threshold below 10^{-8}, results in LMs with roughly the same amount of bi- and trigrams, as shown in Figure 2.

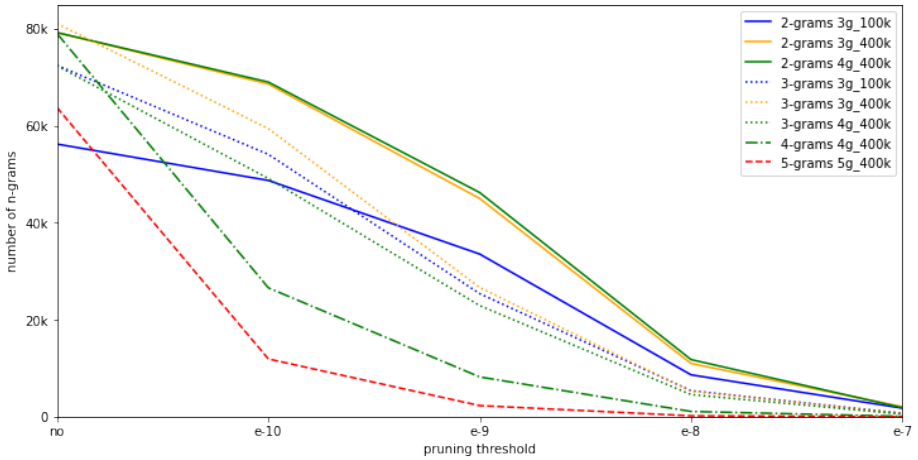


Figure (2) The effect of entropy-based pruning of the number of n-grams for different LMs.

5.2 Optimal pruning strategy

The effect of different pruning strategies on ASR performance is shown in Table 2. We also report the LM sizes in terms of number of n-grams (or equivalently, parameters) and their perplexity (PPL) on the evaluation datasets. The large WER differences among the different evaluation sets is striking and needs some clarification. As already remarked by Demuynck et al. (2008) the development (*Nbest-dev*) and evaluation (*Nbest-test*) parts of the N-Best 2008 benchmark are very different. While *Nbest-test* contains mainly spontaneous speech, dialogues, speaker turns and speech with low quality acoustics, *Nbest-dev* contains more highly intelligible fragments from newsreaders. Also *CGN-dev*, taken randomly from the CGN corpus, contains by design plenty of very short utterance that are hard to recognize by themselves.

Comparing the language models based on *Nbest-test*, we find the largest improvement in performance, a relative gain of almost 6% for the 3-gram, when using a 400k lexicon. We also find the performance of the 400k 3-gram and 400k 4-gram are on par for moderate pruning. Overall we may

conclude that 400k performs consistently better than 100k and 4-gram better than 3-gram, except in extreme pruning circumstances.

LM				*CGN-dev*		*Nbest-dev*		*Nbest-test*	
Lex.	Order	θ	# n-grams	WER	PPL	WER	PPL	WER	PPL
100k	3g	-	129 M	14.02	171	4.44	242	10.99	213
100k	3g	10^{-10}	103 M	14.06	172	4.50	215	11.04	215
100k	3g	10^{-9}	59 M	14.18	177	4.56	252	11.18	220
100k	3g	10^{-8}	14 M	14.53	192	4.69	280	11.83	241
100k	3g	10^{-7}	3 M	15.65	236	5.27	361	12.41	298
400k	3g	-	161 M	12.58	212	3.46	368	10.26	272
400k	3g	10^{-10}	103 M	12.64	214	3.48	370	10.36	274
400k	3g	10^{-9}	72 M	12.75	221	3.58	382	10.58	282
400k	3g	10^{-8}	17 M	13.17	242	3.87	425	11.10	311
400k	3g	10^{-7}	3 M	14.47	302	4.49	564	12.01	390
400k	4g	-	231 M	12.40	198	3.25	337	10.12	257
400k	4g	10^{-10}	103 M	12.48	203	3.27	347	10.15	262
400k	4g	10^{-9}	78 M	12.71	214	3.33	370	10.45	274
400k	4g	10^{-8}	18 M	13.26	247	3.94	437	11.09	314
400k	4g	10^{-7}	3 M	14.62	323	4.65	605	12.42	407

Table (2) Number of n-grams, perplexity (%) and Word Error Rate (%) for different pruned strategies. Note that comparing perplexities only makes sense for a given lexicon size and dataset.

6. Conclusion

ASR technology has advanced greatly over the last 10 years by the introduction of DNN technology. In our work, we find a relative gain in performance of 40% compared to the ESAT 2008 HMM-GMM system. Since the decoding techniques remained similar, we can attribute this improvement to the acoustic modelling. We also conclude that the best strategy for optimizing a language model for a HMM-TDNN system used for single pass decoding in a scenario with memory constraints is using entropy-based pruning of the 400k 4-gram language model up to the desired size, rather than reducing the order. However, the most significant improvement in performance is due to a larger lexicon. This all confirms that adapting lexicon and language model to the use-case is of primary importance. Moreover, we should be able to do this with limited amounts of data if we want to deploy many different small applications. Finally, looking forward, we are developing our hybrid system further in a direction where we meet end-to-end systems halfway. In this we want to replace the phonetic lexicon, which we believe to be the weakest component of the current system. Therefore we are currently investigating lexicons using canonical graphemic transcriptions instead of canonical phonemic transcriptions, where we are using transformers to automatically generate these grapheme-2-grapheme conversions.

References

Demuynck, Kris, Antti Puurula, Dirk Van Compernolle, and Patrick Wambacq (2009), The ESAT 2008 system for N-Best Dutch speech recognition benchmark, *2009 IEEE Workshop on Automatic Speech Recognition & Understanding*, IEEE, pp. 339–344.

Demuynck, Kris, Jan Roelens, Dirk Van Compernolle, and Patrick Wambacq (2008), SPRAAK: an open source "SPeech Recognition and Automatic Annotation Kit", *Proc. Interspeech 2008*, p. 495.

Huijbregts, Marijn, Roeland Ordelman, Laurens van der Werff, and Franciska M. G. de Jong (2009), SHoUT, the University of Twente submission to the N-Best 2008 speech recognition evaluation for Dutch, *Proc. Interspeech 2009*, pp. 2575–2578.

Kessens, Judith and David A. van Leeuwen (2007), N-best: the northern- and southern-dutch benchmark evaluation of speech recognition technology, *Proc. Interspeech 2007*, pp. 1354–1357.

Ko, Tom, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur (2015), Audio augmentation for speech recognition, *Proc. Interspeech 2015*, pp. 3586–3589.

Mertens, Piet and Filip Vercammen (1997), Fonilex manual, *Technical report*, K.U. Leuven. https://lirias.kuleuven.be/retrieve/216423.

Oostdijk, Nelleke (2000), The Spoken Dutch Corpus. Overview and first evaluation, *Proceedings of LREC 2000*, ELRA, pp. 887–893.

Povey, Daniel, Arnab Ghoshal, Gilles Boulianne, Nagendra Goel, Mirko Hannemann, Yanmin Qian, Petr Schwarz, and Georg Stemmer (2011), The Kaldi speech recognition toolkit, *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*.

Povey, Daniel, Gaofeng Cheng, Yiming Wang, Ke Li, Hainan Xu, Mahsa Yarmohammadi, and Sanjeev Khudanpur (2018), Semi-Orthogonal Low-Rank Matrix Factorization for Deep Neural Networks, *Proc. Interspeech 2018*, pp. 3743–3747.

Povey, Daniel, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahremani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur (2016), Purely Sequence-Trained Neural Networks for ASR Based on Lattice-Free MMI, *Proc. Interspeech 2016*, pp. 2751–2755.

Rath, Shakti P., Daniel Povey, Karel Veselý, and Jan Černocký (2013), Improved feature processing for deep neural networks, *Proc. Interspeech 2013*, pp. 109–113.

Röpke, Willem, Roxana Radulescu, Kyriakos Efthymiadis, and Ann Nowe (2019), Training a Speech-to-Text Model for Dutch on the Corpus Gesproken Nederlands, *Proceedings of the 31st Benelux Conference on Artificial Intelligence (BNAIC 2019)*, Vol. 2491 of *CEUR Workshop Proceedings*.

Stolcke, Andreas (2000), Entropy-based Pruning of Backoff Language Models, *Computing Research Repository*. https://arxiv.org/abs/cs/0006025.

Stolcke, Andreas (2002), SRILM — an extensible language modeling toolkit, *Proc. Interspeech 2002*, ISCA, pp. 901–904.

van Leeuwen, David A., Judith Kessens, Eric Sanders, and Henk van den Heuvel (2009), Results of the n-best 2008 dutch speech recognition evaluation, *Proc. Interspeech 2009*, pp. 2571–2574.

Verwimp, Lyan, Brecht Desplanques, Kris Demuynck, Joris Pelemans, Marieke Lycke, and Patrick Wambacq (2016), STON: Efficient Subtitling in Dutch Using State-of-the-Art Tools, *Proc. Interspeech 2016*, pp. 780–781.

Yılmaz, Emre, Henk van den Heuvel, and David van Leeuwen (2018), Acoustic and Textual Data Augmentation for Improved ASR of Code-Switching Speech, *Proc. Interspeech 2018*, pp. 1933–1937.