The CMU-KIT Submissions to the OpenSAT 2017 Evaluation

Florian Metze, Yun Wang, Rajat Kulshreshta, and Markus Müller

CMU-LTI-17-004

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

© 2017, Carnegie Mellon University

THE CMU-KIT SUBMISSIONS TO THE OPENSAT 2017 EVALUATION

TECHNICAL REPORT CMU-LTI-17-004

Florian Metze, Yun Wang, and Rajat Kulshreshta

Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA; U.S.A.
{fmetze|yunwang|rkulshre}@cs.cmu.edu

Markus Müller

Interactive Systems Labs
Karlsruhe Institute of Technology
Karlsruhe; Germany
m.mueller@kit.edu

ABSTRACT

This paper describes the development of the CMU-KIT submissions to the NIST OpenSAT 2017 Evaluation, SAD (not LRL) and ASR (LRL only) tracks. No submissions were made to the KWS track.

The main goals of our submission were (a) to evaluate the performance of a segmentation and non-speech audio event detection algorithm that was originally developed for the Trecvid MED task [1], and (b) to evaluate the performance of our CTC-based end-to-end speech recognition approach [2].

On development data, we achieved DCF=0.091 (SV), DCF=0.040 (PSC), and WER=45.8% (LRL).

***Index Terms*—** OpenSAT evaluation, diarization, speech-to-text, end-to-end speech processing

1. INTRODUCTION

The authors learned of the OpenSAT evaluation only in the afternoon of the last day of the sign-up period, while discussing data requirements for the 2017 JSALT workshop [3], which Carnegie Mellon was organizing, while participating in this evaluation.

The submitted systems thus necessarily present relatively small efforts on top of existing resources. We decided to focus our efforts on SAD for VAST and STT for LRL, while also applying our existing SAD system to the SSSF task without further significant tuning. An existing SAD is available for the LRL task, which had however been tuned for KWS and ASR performance rather than SAD performance, so we decided to re-use it for the LRL submission, but not submit

Despite the limited time available, the performance on the LRL task improved significantly since the Babel BP evaluation in 2013: our single-model system improved from $\sim 57\%$ word error rate (WER) in 2013 to 45.8% WER using automatic segmentation (or 44.0% WER with manual segmentation, both results computed on the DEV data). In 2013, even multi-site system combinations did not break the 50% barrier for the Pashto FLP NTAR condition. We re-used the same automatic segmentation that was developed for the 2013 Babel evaluation for the OpenSAT submission, and made only small tweaks to the n-gram language model, so almost all of the improvements can be attributed to the end-to-end trained acoustic model (see Section 3.1).

While our previous systems used adapted context-dependent HMM-based "hybrid" acoustic models, our current system uses a bi-directional LSTM network trained using the Connectionist Temporal Classification (CTC) [4] loss function. In our setup, a single network is trained towards (context independent) phonetic and character targets in a multi-task training setup, using unadapted lMEL features. During decoding, both outputs are being evaluated with different language models and the hypotheses are combined with ROVER.

Our SAT submissions rely on a large-scale feature extraction that was originally developed for the NIST Trecvid MED task (audio modality), but includes a significantly improved classifier based on recurrent neural networks. The systems were originally developed with information retrieval metrics in mind, rather than speech acitivity detection, but manual inspection on the DEV data indicates good overall performance of the submitted systems. See Section 3.2 for details.

2. DATA RESOURCES

Our systems were trained only on the provided data, specifically, the Full Pashto Language Pack, IARPA-babel104b-v0.4bY, and the provided VAST and SSSF DEV corpora. No external resources were used.

3. ALGORITHMIC DESCRIPTION

3.1. LRL – BABEL Pashto

Our submission uses the Eesen toolkit [2], which combines an acoustic model that has been trained with the Connectionist Temporal Classification [4] (CTC) loss function with

weighted Finite State Transducer (FST) [5] based decoding. Compared to conventional frame-based approaches, this system leads to much faster training, lower memory use, and better performance. We implemented the acoustic model in Tensorflow [6], which lead to further improvements, and allows us to experiment quickly with new architectures or ideas, such as open domain audio-visual speech recognition [7], or character-based (open vocabulary) speech recogition [8]. For further speedup, the LSTM acoustic model uses the cuDNN 5.1 backend.

More information on the Eesen wFST approach can be found in [2]. We experimented with open vocabulary decoding approaches [8], but were not able to further improve WER results, probably due to the small amount and comparably high quality of the Pashto data provided for OpenSAT 2017.

Our acoustic model uses 40 lMEL filter banks as input, and we are adding 3 pitch-related features. We are stacking 3 neighboring frames, for a 129-dimensional input feature vector. The vector is extracted with a step size of 30 ms, in line with other recent DNN work. We are performing data augmentation by extracting three copies of each input utterance, with different offsets of 0, 10, and 20 ms. An acoustic model with 6 bi-directional LSTM layers and 140 cells in each direction performed best in our experiments, and we use an 80-dimensional projection layer between the stacked inputs and the LSTM. The system is trained using the Adam optimizer.

The main novelty for the OpenSAT system is a multi-task training approach: we train a single bi-directional LSTM acoustic model towards both phonetic and (UTF-8) character-based targets. The phonetic dictionary is taken from the training database release, while the characters are given by the set of individual UTF-8 symbols found in the training transcriptions. The system thus predicts 58 phonetic targets and 54 character targets in parallel (in addition to blank) in separate soft-max output layers, while the overall training loss of the system is given by the sum of these two soft-max layers

We trained a 3-gram and a 4-gram language model with IRSTLM [9]. These are then usedto decode the phone posteriors predicted by the acoustic model, after whitening with a "temperature" [10] of 1.5 and scaling down the blank label with 0.7.

We combind the output of the phone-based and character-based decoding approach together using ROVER [11], which proved a simple and effective method to combine the two

3.2. SV – VAST

3.2.1. Feature Extraction

Features are extracted using the OpenSMILE toolkit [12]. The input audio is cut into 25 ms frames shifting by 10 ms. A variety of low-level features are extracted from each frame, such as MFCCs, fundamental frequency, etc. Statistics (e.g. min, max, mean, variance) of these features are computed over 2-second windows shifting by 0.1 s. This yields 6,669 highly correlated dimensions; PCA is applied to reduce them to 50 dimensions. The resulting representation of each audio recording is a sequence of 50-dimensional feature vectors with a frame shift of 0.1 s. This is also the time resolution of the SAD output.

3.2.2. Model Structure

The SAD is performed by a simple bidirectional RNN. It has one single hidden layer with 200 LSTM units in each direction. It has 50 input units and a single output unit, which predicts the probability of speech at each frame.

3.2.3. Model Training

The RNN is trained on the DEV part of the VAST data. The DEV data is partitioned equally into four folds; four models are trained simultaneously, each using three folds as training data and one fold as validation data. The objective function is average cross-entropy, with all frames carrying equal weight. The training algorithm is SGD with a Nesterov momentum of 0.9. Each minibatch contains 5 sequences of 500 frames. The initial learning rate is 0.1, and adjusted according to the cross-validation accuracy. A fifth model is trained simultaneously using all the DEV data as training data, and is used to make predictions on the eval data. This system is described in more detail in Section 3.2 of [13].

3.2.4. Prediction

For each recording of the DEV data, we predict frame-wise probabilities of speech using the RNN whose training data does not contain this recording. For the eval data, we use the RNN trained on all DEV data. A maximum filter of length 7 is applied to smooth the probabilities, which are then binarized with a threshold of 0.960691. The filter length and threshold are optimized on the DEV data.

3.3. PSC – SSSF

3.3.1. Feature Extraction

Same as VAST, as described in Section 3.2.

3.3.2. Model Structure

The SAD is performed by a bidirectional RNN originally trained for sound event detection (SED). It has one single hidden layer with 400 LSTM units in each direction. It has 50 input units and 18 output units in a softmax group. The output units correspond to 17 types of sound events and "background".

System	SUB	DEL	INS	WER
Character	30.7	12.4	3.9	47.0
Phone	29.7	13.0	3.9	46.6
Joint	26.5	14.0	3.5	44.0
Character	30.1	14.4	4.4	49.0
Phone	28.9	15.4	4.2	48.5
Joint	26.1	15.7	4.0	45.8

Table 1: Pashto Babel FLP ASR results (in %) on DEV data, using manual (top half) and automatic (bottom half) segmentation. Character and phone-based systems have virtually identical performance, their combination (using ROVER) results in significant improvement.

3.3.3. Model Training

The RNN is trained on the "noiseme" corpus annotated at CMU [14]. This is a corpus for sound event detection. It contains 464 recording totaling 9.6 h. 17 types of sound events are annotated with onset and offset times. 60% of the corpus is used for training, and 20% for validation. The objective function is average cross-entropy, with all frames carrying equal weight. The training data is augmented four times, by using both channels and two versions of OpenSMILE (1.0.1 and 2.1). The training algorithm is SGD with a Nesterov momentum of 0.9. Each minibatch contains 5 sequences of 500 frames. The initial learning rate is 0.05, and adjusted according to the validation accuracy.

3.3.4. Prediction

For each recording of the DEV data, we predict frame-wise distributions of sound events using the RNN. The total probability of the classes "speech, non-English speech, singing, singing with music, human" are considered as the probability of speech. Even though the evaluation plan postulates that singing should not count as speech, including singing yields better DEV performance. No maximum filter is applied; the threshold is 0.190860 (optimized on the DEV data).

4. RESULTS ON THE DEV SET

4.1. LRL – BABEL Pashto

Table 1 shows the results of our FLP Pashto system. It is important to note that no adaptation or multi-pass decoding strategies are being used. Character- and phone-based system are implemented as a multi-task acoustic model, and decoded with a tri-gram and four-gram language model respectively, before being combined using ROVER. This setup results in an efficient and effective single-pass decoding strategy.

System	DCF	pMiss	pFA
VAST	0.091	0.056	0.194
SSSF	0.040	0.018	0.109

Table 2: Results for SAD on the VAST and SSSF corpora.

4.2. SV – VAST and PSC – SSSF

Table 2 shows the results for speech activity detection on the DEV data. Both systems use largely the same ideas, but have been trained and tuned on different data sets.

5. HARDWARE DESCRIPTION AND TIMING REPORT

The systems were mostly trained on CMU's "Rocks" cluster, using NVidia K20 GPUs (4 per server) and 16-core Intel Xeon E5-2660 (at 2.2 GHz) servers, all with 128 GB of RAM. Some experiments were also run on Nvidia Titan X (Pascal) GPUs (one per server, with 32 Gb).

Training a phone/ character multi-task acoustic model took about 12 h on a Titan X GPU (for $\approx$25 iterations on the Pashto FLP pack, including 5-fold data augmentation). A single-task system without data augmentation takes less then 20 h to train on a single K20 GPU. Feature extraction requires about an hour on an entire machine, while language model training can be achieved within a few minutes.

In the submitted configuration, decoding involves extraction of acoustic model scores on a CUDA-enabled GPU, followed by WFST decoding and extraction of the best hypothesis on CPUs. Extraction of the required state posteriors takes less than 30 mins on a K20 GPU for the 10 h DEV set. This setup enabled us to build and test acoustic models quickly and exclusively on GPUs, while testing many different configurations on a large number of available CPUs. Decoding most models takes less than 0.1 RTF (ten times faster than real-time), so that an end-to-end test run on 10 h of data (with given segmentation) can be turned around in less than one hour on one of our CPU servers and one GPU.

The SAD systems similarly trained in a few hours each. Given the nature of the large-scale feature extraction, it is important to use local storage for keeping temporary and intermediate files whenever possible.

It is thus possible to re-train a system from scratch in less than a day, on a single machine, including all feature extraction and preparation steps. No multi-pass decoding is being used, making ours a very efficient system well suited for rapid prototyping and testing of different setups (e.g. lexicons, script normalizations, data cleaning techniques, etc).

6. CONCLUSION AND FUTURE PLANS

The submitted (single and non-adapted) system is about 5% absolute better than a multi-site system combination from 2013. At the same time, a CTC model is extremely simple to build (our system models context independent phones and characters directly), demonstrating the improvements that deep learning has brought to the speech recognition community over the last few years.

We are currently exploring the following ideas to further improve our CTC-based approach to speech recognition:

- Further tuning and improvements such as batch normalization, improved data augmentation have since improved our performane on other tasks such as Switchboard. We believe most of these techniques could also help on the Babel LRL task.
- We have implemented a wFST-free decoding approach, which rescores the CTC output with an RNN character language model. At present, this approach generally performs a little worse than our wFST baseline, but a vocabulary-free speech recognizer may be useful for recognizing low resource languages with high morphological complexity, as such a system does not suffer from the OOV problem generally associated with closed vocabularies.
- We are implementing adaptation of CTC models (and RNN language models), specifically to visual or auditive context (e.g., if a "car" can bee seen in a video, the ASR is adapted to prefer outdoor acoustics, and vocabulary (strings) from the transport domain.

7. REFERENCES

[1] F. Metze, S. Rawat, and Y. Wang, "Improved audio features for large-scale multimedia event detection," in *Proc. ICME*, Chengdu; China, July 2014, IEEE.

[2] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-End Speech Recognition using Deep RNN Models and WFST-based Decoding," in *Proc. Automatic Speech Recognition and Understanding Workshop (ASRU)*, Scottsdale, AZ; U.S.A., Dec. 2015, IEEE, https://github.com/srvk/eesen.

[3] Carnegie Mellon University, "2017 frederick jelinek memorial summer workshop," https://www.lti.cs.cmu.edu/2017-jelinek-workshop, July 2017.

[4] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with r ecurrent neural networks," in *Proceedings of the 23rd international conference on Machine Learning*. ACM, 2006, pp. 369–376.

[5] M. Riley, C. Allauzen, and M. Jansche, "Openfst: An open-source, weighted finite-state transducer library and its applications to speech and language," in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Tutorial Abstracts*. Association for Computational Linguistics, 2009, pp. 9–10.

[6] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, Software available from tensorflow.org.

[7] A. Gupta, Y. Miao, L. Neves, and F. Metze, "Visual features for context-aware speech recognition," in *Proc. ICASSP*, New Orleans, LA; U.S.A., Mar. 2017, IEEE, Best student paper candidate.

[8] T. Zenkel, R. Sanabria, F. Metze, J. Niehues, M. Sperber, S. Stüker, and A. Waibel, "Comparison of decoding strategies for ctc acoustic models," in *Proc. INTERSPEECH*, Stockholm, Sweden, Aug. 2017, ISCA, Accepted.

[9] M. Federico, N. Bertoldi, and M. Cettolo, "Irst language modeling toolkit, version 5.50. 02: User manual," *FBK-irst, Trento, Italy, November*, 2010.

[10] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," *arXiv preprint arXiv:1612.02695*, 2016.

[11] J. Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER)," in *Proc. Automatic Speech Recognition and Understanding Workshop*, Santa Barbara, CA; U.S.A, 1997, IEEE, pp. 347–354.

[12] F. Eyben, M. Wöllmer, and B. Schuller, "opensmile – the munich versatile and fast open-source audio feature extractor," in *Proc. ACM Multimedia (MM)*, Firenze; Italy, Oct. 2010, ACM.

[13] Y. Wang and F. Metze, "A first attempt at polyphonic sound event detection using connectionist temporal classification," in *Proc. ICASSP*, New Orleans, LA; U.S.A., Mar. 2017, IEEE.

[14] S. Burger, Q. Jin, P. F. Schulam, and F. Metze, "Noisemes: Manual annotation of environmental noise in audio streams," Tech. Rep. CMU-LTI-12-07, Carnegie Mellon University, Pittsburgh, PA; U.S.A., 2012.