INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Model Adaptation and Active Learning in the BBN Speech Activity Detection System for the DARPA RATS program

Damianos Karakos, Scott Novotney, Le Zhang, Rich Schwartz

Raytheon BBN Technologies, USA
{dkarakos,snovotne,lzhang,schwartz}@bbn.com

Abstract

Model adaptation is an important task in many human language technology fields, as it allows one to reduce differences that arise due to various forms of variability. Here, we focus on the speech activity detection (SAD) task, in the context of the DARPA RATS program, where the training data do not cover all channels (transmitter/receiver characteristics) that are encountered at test time. For supervised adaptation, limited manually labeled data from the (novel) channel of interest are used to adapt the model; for unsupervised adaptation, the labels are automatically generated with a baseline model. The modeling is done with long short-term memory neural networks, and we make the case that strong regularization is of paramount importance when adapting such models. Results on two different datasets show that adaptation gives rise to large gains (at least 27related task, that of active learning, is also considered. In active learning, data to be annotated for supervised adaptation are selected automatically, with the ultimate goal of maximizing performance. We investigate an algorithm for active learning that utilizes the output of a SAD decoder and show that it performs significantly better (by 10% relative) than random selection.

Index Terms: speech activity detection, model adaptation, active learning

1. Introduction

When working with real data, one frequently encounters situtations where the test data are not well-matched with the training data used to train a machine learning system. This can be quite problematic, as the performance of the machine learning system could become unpredictably bad, especially when there are multiple degrees of variation. This is what we deal with in this paper, in the context of the speech activity detection (SAD) task, done under the DARPA RATS program [1].

The goal of SAD is to detect the locations in an audio stream that contain speech. These locations may be used to guide manual inspection, or to determine where further downstream processing, such as keyword search or speaker identification, can be performed more reliably.

In RATS, speech is transmitted through "channels", which correspond to unique transmitter/receiver characteristics. Some of these channels are available to annotators, who generate training data for building automatic SAD systems. This entails listening to the audio to determine where speech occurs. However, a number of channels are not available for annotation, or are only available in limited quantities. Since these "novel" channels can be significantly different from the ones encountered in training, the performance drop can be dramatic.

To alleviate this problem, model adaptation is usually done. For example, in speech recognition, unsupervised speaker adaptation is done by estimating a linear transform that makes features less variable across speakers, or, equivalently, GMM models better matched to each speaker. In an unsupervised adaptation setting, this entails using a first-pass transcription of the novel speaker, which is used as noisy training data for estimating the transform. Consistent gains in word-error-rate are obtained with this approach [2]. Of course, when supervised data from a novel speaker are available, the task is less prone to error.

In this paper, we focus on adapting neural networks (NN), specifically, long short-term memory (LSTM) models. LSTMs have performed very sucessfully in a variety of tasks and domains [3, 4, 5], so, it is a natural choice for the problem at hand. NN model adaptation can be done in a variety of ways [6, 7, 8] and we investigate some here.

We focus on three aspects of adaptation: (i) Unsupervised adaptation, where there are no manually annotated data from the novel channel of interest. So, one has to rely on (usually noisy) automatic labels obtained in a first-pass decoding. (ii) Supervised adaptation, where there is limited amount of supervised data from the novel channel of interest. (iii) Supervised adaptation with active learning, where the annotation of the data used for supervised adaptation is done "dynamically", i.e., it can be chosen in a judicious way by the user (usually using automatic means). The end goal is to guide the data selection (based on information, such as the output of a decoder) in a way that maximizes the information content of the data selected.

The performance measure used in this paper is the equal-error-rate (EER), defined as the point on the DET curve where the miss rate is equal to the false-alarm rate (defined below):

$$p_{\text{Miss}} = \frac{\text{total FN time}}{\text{true speech duration}}, \; p_{\text{FA}} = \frac{\text{total FP time}}{\text{true non-speech duration}}, \tag{1}$$

(FN stands for "False Negative" and FP for "False Positive".)

The paper is organized as follows: Section 2 gives an overview of LSTMs, and provides some details about how they are used for SAD. Section 3 gives some background on the problem of model adaptation for neural networks, and discusses techniques for regularizing their training in the presence of limited or noisy data. Section 4 focuses on active learning. Two criteria for active learning, (i) entropy-based and (ii) committee-based, are discussed in detail. Experimental results with model adaptation and active learning are presented in Section 5, and concluding remarks appear in Section 6.

The research presented in the paper was sponsored by the Defense Advanced Research Projects Agency Information Innovation Office (I2O), Program: Robust Automatic Transcription of Speech (RATS) Phases 4 and 5. Issued by DARPA/CMO under Contract No. HR0011-15-C-0038. We would like the thank our BBN colleague, Jan Silovsky, for useful discsussions related to this research.

http://dx.doi.org/10.21437/Interspeech.2016-603

2. Long Short-Term Memory Networks

Long Short-Term Memory (LSTM) neural networks are recurrent networks that are especially useful when training from sequences of data where a long context provides useful information. They do not suffer from the vanishing gradient problem of regular recurrent networks [9], as their unique structure allows them to "remember" the gradient of the error for long durations, and "forget" it when it is not useful [9].

Bidirectional LSTMs are a variant of the above, where the data are scanned in both directions during training and testing. We found out that BLSTMs outperform LSTMs in the SAD task, so, we used them exclusively in this paper.

For SAD, we investigated several BLSTM architectures and input features. Fitting models into GPU memory limited the overall model size, but we experimented with a variety of architectures, such as 30x250x250x2, 30x250x250x250x2 and 30x500ffx500ffx100ffx500x500x2, where '500ff' means 500 feed-forward logistic units. Training was done with stochastic gradient descent using a decaying learning rate and a heldout stopping criterion. Training ran on single NVIDIA Tesla K80 machines using a modified version of the currennt toolkit [10].

Because of computational compexity constraints, we focused on a single architecture with two hdden layers containing 250 nodes each. The per-frame input features were 15-dimensional PLPs with their derivatives (deltas). The targets were binary variables, indicating the presence or absence of speech at each frame.

3. Model Adaptation

In model adaptation, the requirement is to improve the performance of a baseline model as much as possible when applying it to a new domain not encountered in the training data. In the DARPA RATS program, we must adapt to speech transmitted through novel "channels", that is, transmitter/receiver pairs that have unique characteristics such as frequency band, modulation technique, etc. Performance on novel channels is usually very poor with NNs, as they are very good at learning the given training data very well; adaptation techniques that utilize manually annotated or automatically labeled data are thus very important.

There are a number of approaches to neural network adaptation listed in the literature (e.g., see [6, 7, 8]). In all these cases, adaptation is done by running a few training iterations with a pre-trained network as a starting point, using data from the novel domain. Regularization in the training is of importance, as either the adaptation data is limited (supervised adaptation) or noisy (unsupervised adaptation). In our experiments, we have tried various forms of regularization, such as (i) few number of iterations (either fixed or based on a relative improvement threshold), (ii) reduced learning rate, or (iii) updating parameters of one or only a few layers. In the case of unsupervised adaptation, regularization can also be done by subselecting data to use (e.g., throwing away data that are too noisy or for which the decoder is too uncertain and can lead to degradation). In all these approaches, the goal is to learn a generalizable model, i.e., prevent a situation where the model memorizes the new data.

As mentioned above, in the case of unsupervised adaptation, one may want to use automatic labels that are more reliable, e.g., correspond to locations where the confidence of the classification exceeds a threshold. We have done experiments that remove data whose log-likelihood ratios are between a range of percentiles (e.g., 40%/60% and 30%/70%) but have not noticed any gain over using all of the automatic labels.

4. Active Learning

The goal of active learning is to select data to annotate such that the benefit of using that data in a supervised learning (or supervised adaptation) setting is maximal. Of course, this is applicable in cases where the unlabeled data is plentiful and it is impractical (e.g., because of budgetary constraints) to annotate all of it. So, the research question in active learning is all about finding the most "informative" parts of the unlabeled data and doing a careful selection within the given budget. We assume that there is a minimum segment duration d for which it makes sense to have a person listen to the audio and assign labels. We set $d = 5$ seconds in our experiments.

The most obvious way of selecting data for annotation is to do it randomly (or, uniformly); this gives us a baseline with which to compare other more complicated algorithms. To see the benefit of maximum diversification, we considered three variants of random selection: (i) Random across speakers and within each speaker: speakers are randomly picked, and d-duration segments within each speaker are also randomly picked (could be a silence portion of the audio). (ii) Uniform across speakers and random within each speaker: similar to (i), except that the maximum possible number of speakers is picked. This means that as little audio from each speaker as possible is selected, in order to ensure maximum diversity across speakers. (iii) Uniform across speakers and within each speaker: similar to (ii), except that segments within each speaker are spread at uniform intervals so that they cover the audio to the maximum possible extent.

We ran the random methods (i) and (ii) 10 times and averaged the results.

We also ran versions of (ii) and (iii) so that speakers are selected based on a uniform coverage of channels; this allows us to cover all channels equally (so that they all have the same chance to improve) instead of being influenced by how many speakers are assigned to each channel.

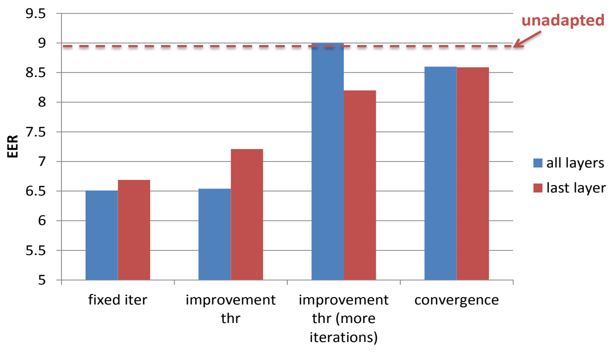
Our results show that all of the above methods perform almost the same, for all amounts of selected audio. Since the uniform method (iii) is better, on average, than both (i) and (ii) (with average EERs 5.30%, 5.29% and 5.28% for (i), (ii), and (iii), respectively), the uniform method is selected in the results below as the baseline.

We also considered a less-diversified alternative of the above, where, instead of picking as many d-long segments as possible, the maximum amount of audio per speaker was chosen (up to the point where all channels had at least some selected audio). This alternative worked much worse (e.g., resulted in EERs which were at least 17% worse than picking as many short segments as possible). This was expected, as picking enough segments to cover enough variations in the audio is an important aspect of the learning.

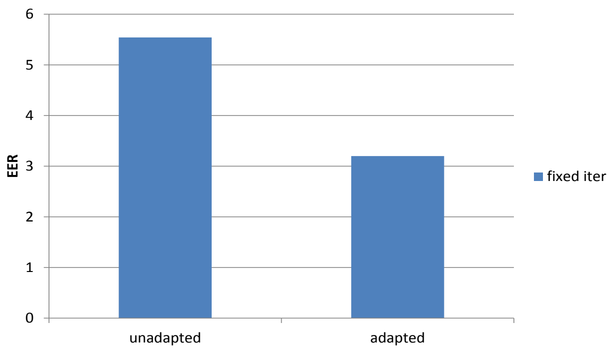
The above algorithms do not use any information about the underlying uncertainty of the model, which is what active learning is supposed to address. So, we implemented two algorithms that rely on one or more decodings of the unlabeled active learning pool of data:

- Entropy-based, where the selected segments correspond to areas of the audio for which the decoder is maximally uncertain about [11, 12]. In mathematical terms, the above algorithm scores each audio frame according to the following formula:

$$S_e(f) = -p_f \log(p_f) - (1 - p_f)\log(1 - p_f) \qquad (2)$$

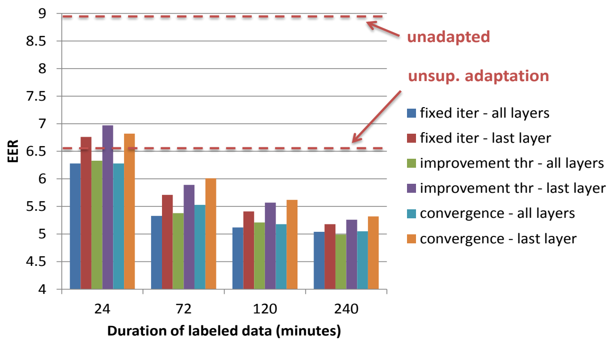


(a)

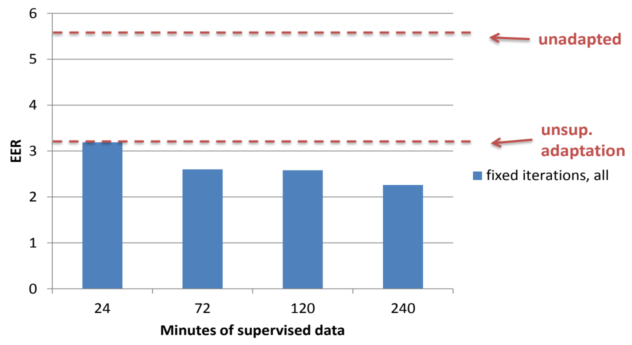


(b)

Figure 1: EER results on (a) Dev-2 and (b) Dev1-p3 with unsupervised adaptation. The amount of unsupervised data was about 70 hours for Dev-2 and 5.4 hours for Dev1-p3.



(a)



(b)

Figure 2: EER results on (a) Dev-2 and (b) Dev1-p3 with supervised adaptation, for various amounts of supervised data.

where p_f is the posterior computed by the NN for frame f.

- Committee-based, where the selected segments correspond to areas of the audio for which multiple decoders[1] maximally disagree about [13]. The formula for scoring each frame is:

$$S_c(f) = \frac{1}{N}\sum_{i=1}^{N} D(p_{f,i}\|\overline{p_f}), \tag{3}$$

where $p_{f,i}$ is the posterior computed by the i-th decoder (NN) for frame f, and $\overline{p_f}$ is the average of those posteriors for frame f. Note that this committee-based method is a modified version of the one in [13][2].

We found in our experiments that the committee method does not offer an advantage over the entropy method, and requires much more computational effort (multiple decodings). Since it is impractical to use it in a realistic scenario, we only consider the entropy method in the rest of the paper.

A few details about how the segments with the highest score (entropy or committee) are selected appear below.

[1]In addition to the baseline model, 2 more models were used: a NN with 2 regular feedforward layers of size 500 each, followed by 2 LSTM layers of size 250 each, and a NN with 3 feedforward layers with sizes 500, 500 and 100, followed by 2 LSTM layers of size 500 each.

[2]The disagreements on hard decisions in [13] were replaced by the distances between the "soft" decisions $p_{f,i}$. This latter approach avoids the problem of having to come up with a threshold for each one of the decoders (which would add one more source of error to the process).

Uniform segmentation: After each frame is scored with one of the algorithms above, segment-level scores are computed (at a 1-second granularity) by averaging the corresponding frame-level scores. Speaker-specific segments are then sorted according to their scores, from high-to-low. Channels and speakers within each channel are then selected in a round-robin fashion, each time selecting the highest-scored segment for that speaker, making sure there is no overlap between the new segment and the previously selected ones. As mentioned above, selected segments have a minimum duration $d = 5$ seconds. One thing to consider in this approach is that, when the amount of audio selected is close to the total amount available, this "greedy" approach could result in "gaps"; that is, short amounts of audio that cannot be selected in the round-robin because they lie between two other selected segments (or, lie at the boundary). To deal with this (again, only when the amount of audio selected is close to the total amount available), we decrease the granularity when computing segment scores (e.g., go from 1 second to 5 seconds); this leaves enough room between two non-adjacent selected segments.

Dynamic programming: A lattice over segmentations is built for the audio of each speaker, allowing each d-second long segment to appear anywhere in the audio, as long as consecutive segments do not overlap. The selection of segments is then done by finding the maximum-scored "path" in this lattice.

As we have found in our experiments, dynamic programming does not give better results than the uniform segmentation. When the amount of audio selected is not close to the maximum amount of audio available, both approaches result in segments that are usually far apart from each other, and the constraints of the dynamic program become less important. On the other hand, when almost all of the audio is selected, the remaining variations of the two approaches are not enough to result in a significant difference in the trained models.

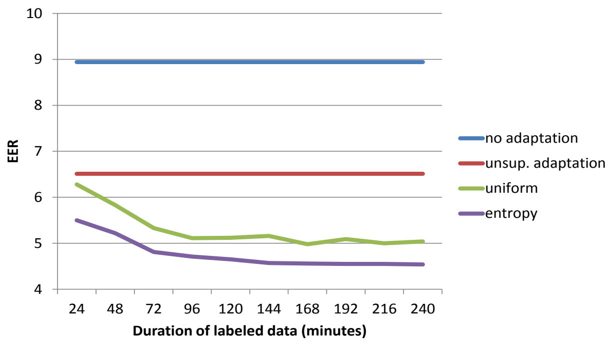
5. Experimental Results

The datasets we used in our experiments were the following: (i) For training a baseline model, we used 1300K+ hours of channels B, D, E, F, G, H, as well as the "clean" channel, obtained through NIST for the OpenSAD-2015 evaluation. (ii) For development purposes (i.e., to tune parameters and obtain best settings) we used the novel channels XA, XH, XI, XK, XN of Dev-2 (about 148.9 hours), obtained through NIST for OpenSAD-2015. (iii) For testing purposes (i.e., to blindly test our models and evaluate performance) we used the novel channels A and C of Dev-1-p3 (about 10 hours). Both development and test set were further split into an "active learning pool" and an "evaluation" set; the former was used in the supervised learning and active learning tasks, for providing manually annotated (supervised) data from the novel channel of interest, while the latter was used to test the models and fairly compute performance.

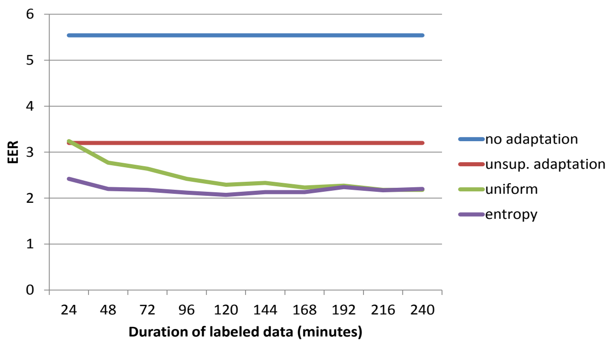
Figure 1(a) shows the EER obtained with unsupervised adaptation, under various regularization approaches: (i) running training for a small, fixed number of iterations (e.g., 2), (ii) using a threshold (e.g., 10%) on the improvement of the cross-entropy to allow early stopping; (iii) same as (ii), but using a more tight threshold (1%) that allows us to run more iterations; (iv) running training to convergence. Variants of the above are also obtained depending on whether only the last layer or all layers of the network are updated. As it is clear from this plot, the most robust approach is to just run a fixed number of iterations of backpropagation, having the baseline model as a starting point. The gain obtained over the non-adapted output is 27% relative on Dev-2 and 42% relative on Dev1-p3. Another interesting observation is that allowing the model to train to convergence can make some of the channels improve, while other channels degrade significantly. This explains why the EER after training to convergence is better than stopping at an earlier iteration when updating all layers. Still, it is much worse than just running very few iterations of training. Figure 1(b) shows the gain from unsupervised adaptation on Dev-1-p3, after applying the best method (fixed number of iterations), selected on the tuning set.

Figure 2 displays EERs for various amounts of supervised data used in the adaptation, along with the EER obtained with the best setting for unsupervised learning. As it is clear from the plots, when the amount of supervision is low, supervised adaptation is on par with unsupervised adaptation, while it improves substantially with more data (reaches 44% on Dev-2 and 58% on Dev-1-p3). As with the case of unsupervised adaptation, using a fixed number of iterations (e.g., 5) and updating all layers of the network yields the best results.

Figure 3 shows learning curves obtained by varying the amount of data used for annotation, in increments of 24 minutes (up to 4 hours). As can be easily seen, for both the development and test data, the entropy method outperforms the uniform method. Note that we also tried to combine this with unsupervised adaptation (first do unsupervised adaptation of the



(a)



(b)

Figure 3: Learning curves on (a) Dev-2 and (b) Dev1-p3, for various algorithms for data selection.

models on the whole test data and then adapt based on the supervised data) but we did not notice any gain over adapting the baseline model.

Note that, since the test part of Dev1-p3 is small (almost 6 hours) it makes sense that all active learning approaches perform similarly when the amount of audio selected is almost the whole audio. The relative gain over the uniform method is 10%, on average over all data durations, for both data sets. This shows that it is clearly beneficial to utilize the output of a decoder when performing active learning, as it guides the learning to focus on areas where it is needed the most.

6. Conclusions

In this paper, we investigated various techniques for adapting a LSTM, for the task of speech activity detection. Our focus was on unsupervised adaptation, supervised adaptation and supervised adaptation combined with active learning. We saw that strong regularization is very important when adapting to new channels, especially when the amount of data is limited or the labels are automatic (noisy). Large gains of 42% relative were obtained on a blind data set with both unsupervised and supervised adaptation. Additional 10% relative gains are obtained with active learning, when using the output of one or multiple decoders to guide the selection. Future work will focus on understanding how different amounts of training affect different channels and how a more flexible form of regularization can prevent the performance on all channels from degrading.

7. References

[1] T. Ng, B. Zhang, L. Nguyen, S. Matsoukas, X. Zhou, N. Mesgarani, K. Veselý, and P. Matejka, "Developing a speech activity detection system for the DARPA RATS program," in *Proc. of Interspeech*, Portland, Oregon, 2012.

[2] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models," *Computer Speech and Language*, vol. 9, no. 2, pp. 171–185, 1995.

[3] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Proc. of Interspeech*, Portland, Oregon, 2012.

[4] F. Weninger, J. Geiger, M. Wöllmer, B. Schuller, and G. Rigoll, "Feature enhancement by deep LSTM networks for ASR in reverberant multisource environments," *Computer Speech and Language*, vol. 28, no. 4, pp. 888–902, 2014.

[5] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proc. of Interspeech*, Singapore, 2014.

[6] H. Liao, "Speaker adaptation of context dependent deep neural networks," in *Proc. of ICASSP*, 2013.

[7] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using I-vectors," in *Proc. of ASRU*, 2013.

[8] S. Thomas, S. Ganapathy, G. Saon, and H. Soltau, "Analyzing convolutional neural networks for speech activity detection in mismatched acoustic conditions," in *Proc. of ICASSP*, Florence, Italy, 2014.

[9] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[10] F. Weninger, J. Bergmann, and B. Schuller, "Introducing currennt: The munich open-source cuda recurrent neural network toolkit," *J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 547–551, Jan. 2015. [Online]. Available: http://dl.acm.org/citation.cfm?id=2789272.2789289

[11] D. Hakkani-Tür, G. Riccardi, and A. Gorin, "Active learning for automatic speech recognition," in *Proc. of ICASSP*, 2002.

[12] A. Venkataraman, Y. Liu, E. Shriberg, and A. Stolcke, "Does active learning help automatic dialog act tagging in meeting data?" in *Proc. of Interspeech*, Lisbon, Portugal, 2005.

[13] S. Argamon-Engelson and I. Dagan, "Committee-based sample selection for probabilistic classifiers," *Journal of Artificial Intelligence Research*, vol. 11, pp. 335–360, 1999.