CHASING THE METRIC: SMOOTHING LEARNING ALGORITHMS FOR KEYWORD DETECTION

Oriol Vinyals[1,2]*, Steven Wegmann*[2]

[1]University of California, Berkeley, [2]International Computer Science Institute

ABSTRACT

In this paper we propose to directly optimize a discrete objective function by smoothing it, showing it is both effective at enhancing the figure of merit that we are interested in while keeping the overall complexity of the training procedure unaltered. We looked at the task of keyword detection with data scarcity (e.g., for languages for which we do not have enough data), and found it useful to optimize the Actual Term Weighted Value (ATWV) directly. In particular, we were able to automatically set the detection threshold while improving ATWV by more than 1% using a computationally cheap method based on a smoothed ATWV on both single systems and for system combination. Furthermore, we did study additional features to refine keyword candidates which were easy to optimize thanks to the same techniques, and improved ATWV by an additional 1%. The advantage of our method with respect to others is that, since we can use continuous optimization techniques, it does not impose a limit in the number of parameters that other discrete optimization techniques exhibit.

Index Terms— Keyword Detection, Optimization, Discrete Metrics

1. INTRODUCTION

Keyword detection is the task of deciding whether a certain word appears in an audio segment. It can be seen as a subproblem of speech recognition, and this is what motivated the approach that ICSI took on the IARPA Babel program [1]. The goal of the program "is to rapidly develop speech recognition capability for keyword search in a previously unstudied language, working with speech recorded in a variety of conditions with limited amounts of transcription." This paradigm contrasts with other keyword detection approaches where the number of keywords to detect is limited, and the data is not scarce (e.g., to activate a device with a specific keyword in a commercial system).

Although in a detection task one typically cares about the precision and recall of the detector, since in Babel we are given a long list of keywords with very different characteristics (e.g., common words and extremely rare words), averaging across each keyword does not seem to be the optimal metric. Instead, they define the Actual Term Weighted Value (ATWV), which is a weighted average (with keyword specific weights based on how frequent the keyword is in the test bed) of a metric related to probability of misses and false alarms. In particular, for a posting entry i with keyword k, TWV(k,i) is:

$$\mathrm{TWV}(k,i) = \mathbf{I}(s(i) > th(k)) \left(\frac{1}{N_k}\mathbf{I}(i = hit) - \frac{\beta}{T - N_k}\mathbf{I}(i = FA) \right)$$

where $\mathbf{I}(.)$ is the step function, $s(i)$ is the score for the i-th entry, $th(k)$ is the keyword specific threshold for keyword k, $\beta = 999.9$, T is the total evaluation time in seconds, and N_k counts how many times keyword k occurs in the current dataset. This metric basically counts the posting entry only if the score is above a certain (keyword specific) threshold. If the posting entry i was a hit, it gives a reward (which is larger for rare keywords), and if it was a false alarm it has a cost (that is approximately constant since T is typically much larger than N_k). ATWV is then simply:

$$\mathrm{ATWV} = \frac{1}{|K|} \sum_k \sum_{i_k} \mathrm{TWV}(k, i_k) \quad (1)$$

where i_k goes through every index in the posting list that contains keyword k as a candidate, and $|K|$ is the total number of keywords. Note that ATWV is upper bounded by 1.

In this paper, we propose to smooth ATWV based on recent work [2] which suggests that direct optimization of a smoothed discrete figure of merit (such as ATWV) is a cost effective method to achieve close to optimal solutions.

2. THE ICSI SYSTEM AND BABEL

The ICSI system can be split into two major subsystems: standard speech recognition, and the keyword search system. We describe them in the following two sections:

2.1. The recognition system

The Kaldi speech recognition toolkit [3], along with the TNet[1] toolkit, were used for recognition and lattice generation.

The full system description can be found in [4], and an updated version in [5], but here we summarize it for completeness.

The ICSI system uses 13 MFCCs as primary features after cepstral mean subtraction. We extract pitch and probability-of-voicing (PoV) features using a sub-band autocorrelation classification, SAcC [6]. These two features are smoothed, interpolated, and then pasted with the cepstral features to form a 15-dimensional feature vector. While ICSI uses Δ and $\Delta\Delta$s for early systems in the initialization, we use a variant of HLDA features for the final systems. The LDA transformation takes as input a context of 7 spliced static MFCC vectors and is trained using the context dependent states as targets;

Supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense US Army Research Laboratory contract number W911NF-12-C-0014. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.

[1]http://speech.fit.vutbr.cz/software/neural-network-trainer-tnet

the features are projected down to 30 dimensions. During training, this LDA matrix is composed with global MLLT matrices as well as speaker dependent fMLLR matrices [7].

We also use 30-dimensional tandem bottleneck (BN) features [8] that are obtained using a hierarchical NN [9]. See [4] for more details. For the tandem features, the system pastes combinations of cepstral, pitch and bottleneck features together to form the tandem feature vector. Note that HLDA is applied to the cepstral part of the features only, while MLLT and fMLLR are applied to the combined feature stream.

Following this is an HMM system with a standard continuous acoustic model where the emission probabilities of the context dependent states are derived from subspace Gaussian mixture models (SGMM) [10]. We use a 3-gram language model on the training transcripts, and apply Kneser-Ney smoothing and interpolated counts [11, 12].

2.2. The KWS system

The KWS system consists of three steps: i) converting recognition lattices to indexes, ii) searching the indexes for a given KW and constructing a posting list, iii) and setting the KW-specific detection threshold in order to optimize ATWV. These are further described below. Note that the KWS systems described here are entirely word-based, i.e., the ICSI system does not combine the word-based search with a separate subword-based search in order to handle out-of-vocabulary (OOV) KWs, e.g., as in [13, 14].

i. ICSI uses "lattice-tool" from SRILM [11] to convert lattices to word-level indexes. We set the lattice-tool parameter that controls how far apart in time two occurrences of a word have to be in the lattice before we consider them separate entries in the index to be 0.1 sec.

ii. For single word KWs the posting list is simply all of the occurrences of the KW sorted by their posterior probabilities. To construct the posting list for a multi-word KW ICSI follows [13]: the individual words are first retrieved from the index in the correct order with respect to their start and end times, but occurrences are discarded when the time gap between adjacent words is more than 0.5 seconds. The surviving occurrences are assigned a detection probability equal to the minimum of the individual word probabilities.

iii. To determine the detection threshold for a given KW we used an empirical threshold to approximately maximize TWV for each keyword. However, much of our system and improvements produce a threshold-less system, so this step is omitted in the experiments reported in Section 3.2 and onwards. However, for the baselines in the Standard Metrics section, this step is still performed as it produces much better thresholds than setting them to a fixed value for all keywords.

2.3. The Data

The audio data in each language is conversational telephone speech recorded in a variety of environments and handset types. There were several languages that program participants worked with, and in this paper we used Pashto (release IARPA-babel104b-v0.4bY), Vietnamese (release IARPA-babel107b-v0.7), and Bengali (release IARPA-babel103b-v0.3). Each language comes with about 80 hours of transcribed training data (Full Language Pack), a pronunciation lexicon that covers the words in the training data, and a 10 hour development test set. All of the results in this paper report KWS results using the keywords (KW) provided for the evaluation. Also, when we report results on the evaluation data we will be restricting the results to the subsets, called "eval-part1", where the ground truth was released to participants: a 15 hour subset in the case of Vietnamese and 5 hour subsets for the rest of the languages. We have also tested our system on a new language released after the first year of Babel, Bengali, to show that the approach generalizes.

3. DISCRIMINATIVE POSTING REFINEMENTS

Our contributions to the ICSI Babel system have been mostly about modifying the posting lists by changing the score from each keyword candidate, trying to achieve better overall performance. We have considered several features (besides the posting entry) to enhance the posterior estimate of the detected posting entry (i.e., keyword and time segment pair). Here is an example of a very short posting list:

```
CONV#14 house 1.3s-1.9s 0.56
CONV#15 house 6.3s-6.8s 0.12
CONV#15 castle 0.3s-1.2s 0.94
```

Note that each keyword candidate has a time, score (from lattice-tool) and conversation id associated with it.

The features that we considered are:

- Acoustic Features: features inspired by the sparse coding features presented in [15], by taking a window (centered around the keyword candidate)
- Acoustic Quality Features: features such as signal to noise ratio (SNR) extracted with SNREval[2] and speaking rate [16]
- Keyword Specific Features: features such as the frequency or length of a keyword
- Lattice Specific Features: since posting lists are derived from lattices, we used arching measurements around the keyword as an additional hint for confidence (the score output in step ii. is already a good estimate)
- Neural Network Posterior Features: neural networks are used to predict phone states per frame. We took the entropy around the keyword of these posterior estimates as another proxy to confidence

All the features above were computed, whenever possible, at the posting list level (i.e., only on the span of the keyword candidate), at the utterance level (i.e., on the span of the speech utterance given by the speech/non-speech detector on the whole conversation side), and at the global level (i.e., considering the whole conversation).

Initially, we attempted to learn standard classifiers using these features. In particular, we considered both linear classifiers (logistic regression and SVM), as well as non-linear classifiers (NN). We built a simple dataset where, for each entry in the posting list, we assign a corresponding label whether the entry was a hit (i.e., the keyword was indeed in the ground truth), or not. This binary classification task was very unbalanced since the posting lists were typically very biased to produce a large number of false alarms. In the following subsection we define some baselines and how standard classifiers performed in this task.

[2] http://labrosa.ee.columbia.edu/projects/snreval/

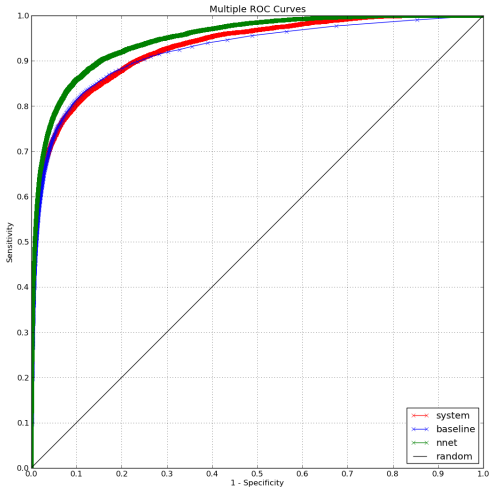

Fig. 1. Receiver operating characteristic for Pashto on the training data using ICSI system (baseline, in blue), our logistic regression system (system, in red) and a neural network (nnet, in green).

3.1. Optimizing Standard Metrics

In Table 1 we show accuracy, likelihood, and ATWV for the training data, which consists of about 400K posting entries for Pashto. The baseline system uses as features the scores that are already present in the posting lists (and that are used by the ICSI system) without any additional features, and trains a simple logistic regression (with one parameter and a bias). The rest used all the features available, but SVM used hinge loss (optimizing accuracy), and Logistic Regression and Neural Network used cross entropy (optimizing likelihood).

Note that, in general, accuracy and likelihood are not great measures of ATWV (which we further discuss in the following section), and ICSI and Baseline systems, even though they use the same features, do not exhibit the same performance. This can be explained because the features get scaled by the logistic regression model (Baseline) which has a negative effect on the following module that sets the threshold for each keyword.

As another example of how different ATWV is with respect to standard binary decision tasks, Figure 1 shows the Receiver Operating Characteristic (ROC) curve for the ICSI system (blue), the Logistic Regression (red) and the Neural Network (green). It is clear that the green curve is better at any point of the precision / recall curve, but in terms of ATWV both systems perform similarly. The reason, which is also described in detail in a recent thesis from one of the members of the ICSI team [17], is that rare keywords should be regarded higher than common keywords (as they have higher TWV). Instead of trying an ad-hoc method (such as biasing our dataset) to achieve this, we chose to directly optimize ATWV, which is described in the following section.

3.2. Optimizing ATWV

As previously discussed, instead of optimizing accuracy or likelihood, we will optimize ATWV, which is the object of interest for Babel. Even though we could use discrete optimization techniques such as the Powell method [18] as used recently in [19], we prefer smooth objective functions as these are faster to optimize, and are not limited to linear models with few parameters. As a result,

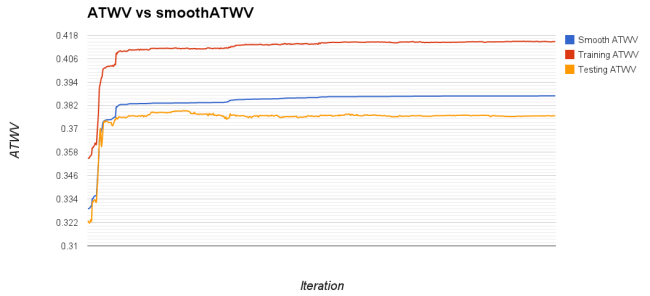

Fig. 2. Smooth ATWV as the training progresses, as well as ATWV on both training and testing for Pashto for the system with only the score feature.

we can use more sophisticated regression models such as neural networks. Recall from equation 1 that ATWV is a discrete objective as it is a function of thresholding our decision function for sample i, $s(i)$. Thus, we take the simple approach of smoothing the unit step with the sigmoid function, so that:

$$\mathbf{I}(x > 0) \approx \sigma(\mathcal{S}x) = \frac{1}{1 + \exp(-\mathcal{S}x)}$$

which converges to the left hand side as $\mathcal{S} \to \infty$. Recent work [2] suggests that optimizing a non-convex approximation of the discrete function is almost always better than to find a convex relaxation (such as replacing $I(x > 0)$ with the hinge loss). In fact, the SVM approach in the previous section (with some modifications) would be close to doing such convex relaxation. But, since the dataset is quite small (less than a million samples), we found the direct optimization of a smooth version of ATWV more compelling. As a result, we define from equation 1:

$$\text{smoothATWV} = \frac{1}{|K|} \sum_k \sum_{i_k} \sigma(\mathcal{S}(s(i_k) - th(k)))$$

$$\left(\frac{1}{N_k} \mathbf{I}(i_k = hit) - \frac{\beta}{T - N_k} \mathbf{I}(i_k = FA) \right)$$

which now can be differentiated with respect to $s(i_k)$, and further to the parameters with chain rule (assuming we parameterize $s(i_k; \boldsymbol{\theta})$, which will be referred to as the model). As a result, we define a new objective function in which we can find the optimal parameters $\boldsymbol{\theta}$ that optimize ATWV (or, rather, an approximation to it). Lastly, since the model typically has a bias which can partially absorb $th(k)$, and having a model that has a common threshold that is keyword independent has certain advantages (e.g., for system combination), we set $th(k) = 0.5 \; \forall k$. This also has the advantage of eliminating step iii in Section 2.2.

With this approach, and setting $\mathcal{S} = 10$, we ran L-BFGS on two models: a log linear model, and a neural network. In both cases, the input for the posting entry i are the features $\mathbf{x}_i$, and the parameters of the model are $\boldsymbol{\theta}$, resulting in $s(i; \boldsymbol{\theta}) = f(\mathbf{x}_i, \boldsymbol{\theta})$ with $f(.)$ being either $\sigma(.)$ (for the log linear model) or a neural network with one binary output. Figure 2 shows the goodness of the approximation of the smooth ATWV (blue vs. red curve), as well as showing that the whole framework can indeed learn a better ATWV than the baseline of 0.4010.

In Table 2 we show the results of our experiments. First, note that our neural network, even though it fits the training data better

System	Features	Accuracy	Negated Log Likelihood	ATWV
Majority Voting	None	97.25%	N/A	0.0
ICSI	Score	97.31%	N/A	0.4010
Baseline (Log. Reg.)	Score	97.79%	0.08	0.3989
SVM	+KW features +Acoustic features	98.21%	0.11	0.3976
Logistic Regression	+KW features +Acoustic features	98.09%	0.07	0.4005
Neural Network	+KW features +Acoustic features	97.98%	0.06	0.4012
Upper Bound	Oracle	100.0%	0.0	0.71

Table 1. Several models and metrics measured on the Pashto training set.

System	Features	ATWV (train)	ATWV (validation)
ICSI	Score	0.4010	0.3685
Log Linear Model	Score	0.4065	0.3752
Log Linear Model	+KW features	0.4118	0.3799
Log Linear Model	+KW features +Acoustic features	0.4138	**0.3835**
Neural Network	+KW features +Acoustic features	0.4297	0.3764

Table 2. Several models and features when directly optimizing ATWV in the Pashto BABEL dataset.

than the simpler log linear model, overfits the data and the testing ATWV is not competitive (even though it is better than the original ICSI system). Secondly, even when considering only the features used in the ICSI system (i.e. step ii of Section 2.2), the resulting system is superior (0.3752 vs. 0.3685). This means that, even though a lot of effort has been put trying to find the optimal threshold for the score coming from the lattices, the automated system can do a better job figuring out how to optimize ATWV directly.

As a last note, even when applying the Log Linear Model with Score features trained on Pashto on a different language (Bengali), the ATWV of the resulting system is also better. The Bengali ICSI system has an ATWV of 0.3123, whereas transferring the model from Pashto to Bengali yielded a slight improvement with an ATWV of 0.3133. Also, when optimizing the model using Bengali, we were expecting similar gains than with Pashto. Indeed, the ATWV with all the features was of 0.3287, a significant improvement with very little computational overhead. Lastly, we also performed an experiment using the limited language pack (i.e., less training data), and on Bengali we obtained an improvement from 0.1287 ATWV to 0.1476.

4. FURTHER IMPROVEMENTS OF ATWV

In this section, we describe current and future efforts that would further improve ATWV. Since we found an efficient method to directly optimize this metric, we explored how we could incorporate this in other parts of the system. In fact, part of what we already were doing was to optimize ATWV on single scalar values such as language or acoustic modeling scales using a simplex method which did not require to smooth the ATWV, and were already improving it by as much as 10% relative.

4.1. System Combination using Optimized ATWV

Another idea we tried that seems to help in terms of ATWV is system combination. Although it is out of the scope of this paper to review system combination in the Babel project, the main idea is to replace the human designed heuristic to combine systems by one that would be learned with the procedure described in this paper. The way we generate systems is by randomly selecting subsets of front-end features, and by merging posting lists of such systems. If an entry of system A overlaps with an entry of system B, the heuristic we found to work was to add the scores. Our method found the optimal way of linearly combining scores of two (or more) systems with little computational overhead (certainly none when comparing to training the full system).

System	ATWV (validation)
ICSI A	0.3870
ICSI B	0.3697
A+B Heuristic	0.4446
A+B Our System	**0.4534**

Table 3. System combination on the Vietnamese development data comparing human heuristics and our method to directly optimize ATWV.

Results using Vietnamese can be found in Table 3. We use a different language than in previous sections to further show generalization of our technique across languages. Indeed, even though we tested several heuristics and converged to keeping the sum of two scores when segments from different systems overlap, learning how to linearly combine the systems (or, potentially, adding other features such as the max of the scores) yielded an improvement of almost 1% (the weights found by our system favored ICSI A, as that system had better ATWV).

5. CONCLUSIONS

We conclude this paper by summarizing it. As it has been already shown in the speech recognition community, optimizing errors that are closely related to the figure of merit that we care about (e.g., WER in the case of ASR) is useful. Thus, when we develop a new system with another metric that we care about, we generally should directly optimize it since this typically outperforms optimizing other related metrics that do not necessarily correlate well with our figure of merit. In this paper we proposed an efficient method to change the objective function aiming to optimize the smoothed figure of merit (in the keyword detection Babel project, ATWV), at the cost of making the objective function non-convex. We do not see this as a real limitation since some of our models (e.g., deep architectures) are already non-convex.

6. REFERENCES

[1] "IARPA Babel Program - Broad Agency Announcement (BAA)," `http://www.iarpa.gov/Programs/ia/Babel/solicitation_babel.html`, 2011.

[2] T. Nguyen and S. Sanner, "Algorithms for direct 0–1 loss optimization in binary classification," in *International Conference on Machine Learning (ICML)*, 2013, pp. 1–9.

[3] D. Povey, A. Ghoshal, et al., "The Kaldi Speech Recognition Toolkit," in *Proc. ASRU*, 2011.

[4] K. Riedhammer, Van Hai Do, and J. Hieronymus, "A study on LVCSR and keyword search for tagalog," in *Proc. Interspeech*, 2013.

[5] S. Wegmann, A. Faria, A. Janin, K. Riedhammer, and N. Morgan, "The tao of atwv: Probing the mysteries of keyword search performance," in *ASRU*, 2013.

[6] B.S. Lee and D. Ellis, "Noise robust pitch tracking using subband autocorrelation classification (SAcC)," in *Proc. Interspeech*, 2012.

[7] M. J. F. Gales, "Maximum likelihood linear transformations for hmm-based speech recognition," *Computer Speech and Language*, vol. 12, pp. 75–98, 1998.

[8] F. Grezl, M. Karafiat, S. Kontar, and J. Cernocky, "Probabilistic and bottleneck features for lvcsr of meetings," in *Proc. ICASSP*, 2007, pp. 757–760.

[9] F. Valente, J. Vepa, C. Plahl, C. Gollan, H. Hermansky, and R. Schlüter, "Hierarchical neural networks feature extraction for lvcsr system," in *Proc. Interspeech*, 2007, pp. 42–45.

[10] D. Povey, L. Burget, M. Agarwal, P. Akyazi, F. Kai, A. Ghoshal, O. Glembek, N. Goel, M. Karafiát, A. Rastrow, R. Rose, P. Schwarz, and S. Thomas, "The subspace Gaussian mixture model - A structured model for speech recognition," *Computer Speech and Language*, vol. 25, pp. 404–439, 2011.

[11] A. Stolcke, "SRILM – an extensible language modeling toolkit," in *Proc. ICSLP*, 2002, pp. 901–904.

[12] R. Kneser and H. Ney, "Improved backing-off for n-gram language modeling," in *Proc. ICASSP*, 1995, pp. 181–184.

[13] D. R. H. Miller, M. Kleber, C.-L. Kao, O. Kimball, T. Colthurst, S. A. Lowe, R. M. Schwartz, and H. Gish, "Rapid and accurate spoken term detection,," in *Proc. Interspeech*, 2007, pp. 314–317.

[14] B. Kingsbury, J. Cui, X. Cui, M. J. F. Gales, K. Knill, J. Mamou, L. Mangu, D. Nolan, M. Picheny, B. Ramabhadran, R. Schluter, A. Sethy, and P. C. Woodland, "A high-performance Cantonese keyword search system," in *Proc. ICASSP*, 2013, pp. 8277–8281.

[15] O. Vinyals and L. Deng, "Are sparse representations rich enough for acoustic modeling?," in *Proc. Interspeech*, 2012.

[16] N. Morgan and E. Fosler-Lussier, "Combining multiple estimators of speaking rate," in *Proc. ICASSP*, 1998.

[17] R. Prabhavalkar, *Discriminative Articulatory Feature-based Pronunciation Models with Application to Spoken Term Detection*, Ph.D. thesis, 2013.

[18] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press, New York, NY, USA, 2007.

[19] D. Karakos, R. Schwartz, S. Tsakalidis, L. Zhang, S. Ranjan, T. Ng, R. Hsiao, G. Saikumar, I. Bulyko, L. Nguyen, J. Makhoul, F. Grezl, M. Hannemann, M. Karafiat, I. Szoke, K. Vesely, L. Lamel, and V. Le, "Score normalization and system combination for improved keyword spotting," in *ASRU*, 2013.