WHOLE WORD DISCRIMINATIVE POINT PROCESS MODELS

Aren Jansen

Johns Hopkins University
HLT Center of Excellence, Center for Language and Speech Processing
Baltimore, MD 21211

ABSTRACT

This paper introduces a discriminative extension to whole-word point process modeling techniques. Meant to circumvent the strong independence assumptions of their generative predecessors, discriminative point process models (DPPM) are trained to distinguish the composite temporal patterns of phonetic events produced for a given word from those of its impostors. Using correct and incorrect word hypotheses extracted from large vocabulary recognizer lattices, we train whole-word DPPMs to provide an alternative set of acoustic model scores. Using solely the timing of sparse phonetic events, DPPM scores exhibit comparable discriminative power to those produced by a state-of-the-art acoustic model built using the IBM Attila Speech Recognition Toolkit. In addition, the inherent complementarity of frame-based and event-based models permits significant improvements from score combination.

Index Terms— point process model, speech recognition, discriminative training

1. INTRODUCTION

The majority of modern speech recognition technology relies on a bottom-up strategy of constructing a set of subword unit models and then using a pronunciation dictionary to construct word-level hidden Markov models in terms of subword unit states. However, due to long range context dependencies, it is reasonable to expect that directly modeling entire words may permit a more accurate and robust decoding of the speech signal [1]. In this paper, we continue research into the development of whole word point process models (PPM) [2][3]. The PPM framework deviates from traditional frame-based methods by abandoning dense vector time series representations in favor of sparse temporal point patterns of salient acoustic or phonetic events. Then, instead of representing words using hidden Markov models with phonetic states, PPMs explicitly model the temporal statistics of the events within each word as a whole.

In previous studies with a generative PPM framework, it has been demonstrated that the timings alone of sparse phonetic events (i.e. discarding all event confidence measures) provide sufficient information to recognize words with comparable accuracies and improved robustness to equivalent frame-based baselines [3]. However, the generative PPM framework relied on strong event- and phone-level independence assumptions that can be more easily circumvented with a suitable discriminative model. In this paper, we consider whole-word kernel-based classifiers that take as input temporal patterns of phonetic events and produce word confidence scores.

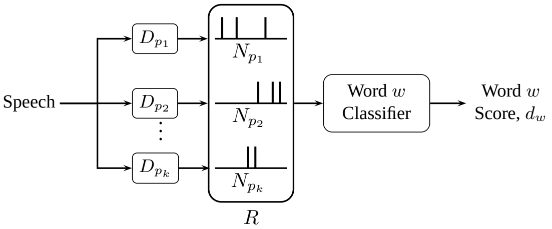

Fig. 1. High-level model architecture.

Like any discriminative model, our proposed discriminative point process models (DPPM) require presegmented positive and negative examples for training. Thus, as in other discriminative training schemes for speech recognition, we must rely on a first pass over the training data using a generative model to provide candidate hypotheses, which can be labeled against a forced alignment of the reference transcript. In this study, we extract our training examples from large vocabulary recognizer word lattices to train whole-word DPPMs, and use them to provide an alternative set of lattice arc acoustic confidence scores. In the past, alternative scores have been demonstrated useful for a range of speech applications, including spoken term detection [4], utterance verification for spoken dialog systems [5], and lattice rescoring for large vocabulary recognition [6].

We find the discriminative power of DPMMs to be comparable to word posteriors produced by a state-of-the-art, discriminatively trained acoustic model built with the IBM Attila Speech Recognition Toolkit [7]. Moreover, our DPPM scores are produced with an entirely distinct modeling architecture, which introduces a complementarity with the baseline system that allows further gains from score combination. In a subsequent effort, significant word error rate reductions were observed when DPPM scores were integrated with the baseline recognizer using the segmental conditional random field (SCARF) framework, as discussed in [8].

2. DISCRIMINATIVE PPM FRAMEWORK

The point process word modeling framework, as presented in [2], consists of two primary components (see Figure 1): a set of detectors $\{D_p\}_{p\in\mathcal{P}}$ for the set $\mathcal{P}$ of English phones, and a set of word detectors $\{d_w\}_{w\in\mathcal{W}}$ for a lexicon $\mathcal{W}$. Each phone detector D_p is tasked with computing a point pattern $N_p = \{t_1, t_2, \ldots, t_{n_p}\}$ on the positive real line, comprised of those points in time that phone p is most clearly expressed. The composite set of point patterns $R = \{N_p\}_{p\in\mathcal{P}}$ defines a sparse point process representation for modeling. A word model for each $w \in \mathcal{W}$ is used to map subsets $R_I = R|_I$ of R restricted to a candidate interval $I = (a, b]$ into scores $d_w(R_I)$ that takes high values when I contains exactly w.

The late Partha Niyogi contributed to this work and would be a co-author if he were able to provide his consent. This paper is dedicated to him.

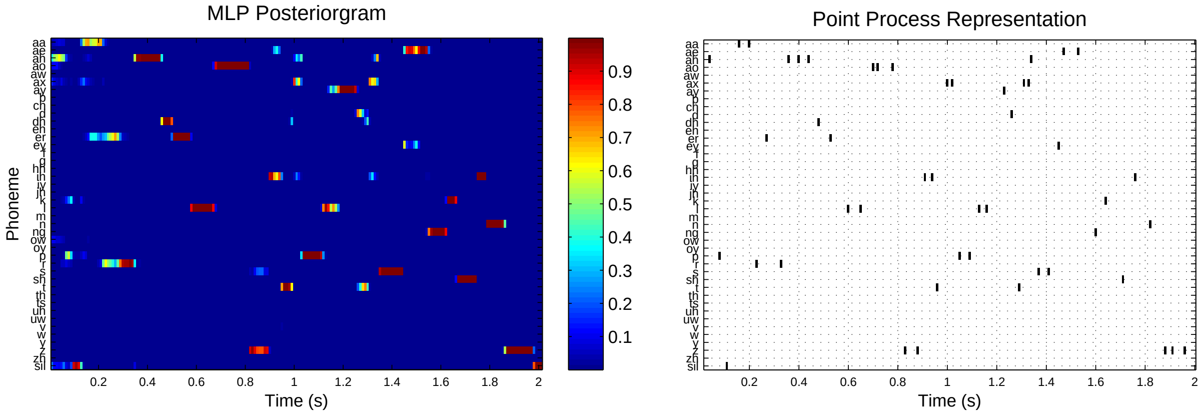

Fig. 2. An example MLP posteriorgram and the corresponding phonetic point process representation ($\delta = 0.5$).

2.1. MLP-Based Phone Detectors

Each phone detector D_p is defined as the composition of two operations. First, we apply a phone-dependent function $g_p : \mathbb{R}^m \rightarrow \mathbb{R}$ to each frame of some m-dimensional feature vector time series $X = x_1 \dots x_T$ to produce a phone detector time series $\{g_p(x_1), \dots, g_p(x_T)\}$ that should take high values only when phone p is present. In prior work [2], we used a monophone Gaussian mixture model-based acoustic model to define each g_p. In this study, we consider instead a discriminatively trained, multilayer perceptron (MLP)-based English monophone model [9, 8], which produces a phonetic posteriorgram of dimension $|\mathcal{P}| = 42$. The posteriorgram is the vector time series $Y = y_1 \dots y_T$, where each y_i is defined as the posterior distribution over the phone set for the ith observation, $y_i = \langle P(p_1|x_i), \dots, P(p_{|\mathcal{P}|}|x_i)\rangle \in \mathbb{R}^{|\mathcal{P}|}$, as computed by the MLP. Given Y for an utterance, the detector time series is defined as the posterior trajectory for phone p such that $g_p(x_i) = y_i[p] = P(p|x_i)$. Second, we apply a thresholded peak finding function that computes the point pattern N_p as the times of all local maxima of g_p that exceed some threshold δ. The individual point patterns collected into a set $R = \{N_p\}_{p\in\mathcal{P}}$ defines our point process representation. Figure 2 shows an example posteriorgram (computed from the MLP monophone acoustic model) and corresponding point process representation.

2.2. Discriminative Word Models

Given the point process representation defined above, we need to construct suitable models for each word in terms of the temporal statistics of the phone events. To accomplish this discriminatively, we consider the learning framework of kernel machines, which has precedent in word level acoustic modeling [10].

Kernel Machines: In the binary classification setting, we are provided a collection of N labeled points in a d-dimensional vector space, $\{x_i, y_i\}_{i=1}^N$, where each $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$. The goal is to learn a function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ such that $\operatorname{sgn}(f(x_i)) = y_i$ as frequently as possible without overfitting. The kernel machine framework attempts to achieve this by solving the optimization problem

$$f^* = \arg\min_{f\in\mathcal{H}_K} \frac{1}{N}\sum_{i=1}^{N} V(y_i, f(x_i)) + \gamma\|f\|_K^2, \qquad (1)$$

where V is some loss function, $\mathcal{H}_K$ is a representing kernel Hilbert space (RKHS) for the kernel function $K : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$, γ is a regularization parameter to control against overfitting, and $\|\cdot\|_K$ denotes the RKHS norm. Common loss functions include (i) the hinge loss, $V(y, f(x)) = \max(0, 1 - yf(x))$, which gives rise to support vector machines, and (ii) the square loss, $V(y, f(x)) = [y - f(x)]^2$, which gives rise to regularized least squares (RLS).If K is a symmetric, positive semi-definite kernel, then it follows by the representer theorem that the solution to Equation 1 can be written as the expansion $f^*(x) = \sum_{i=1}^N \alpha_i K(x_i, x)$, where the $\alpha_i \in \mathbb{R}$ are the new parameters to be learned from the data. In the case of RLS, a simple closed form solution exists and is given by $\alpha = (\mathbf{K} + \gamma N\mathbf{I})^{-1}\mathbf{y}$, where $\alpha = [\alpha_1 \dots \alpha_N]^T$, $\mathbf{y} = [y_1 \dots y_N]^T$, and $\mathbf{K}$ is the $N \times N$ Gram matrix with elements $\mathbf{K}_{ij} = K(x_i, x_j)$.

***Extension to PPMs*:** Our goal is to apply these kernel machine methods to building word classifiers using the above point process representation. Here, training examples take the form $\{R_i, y_i\}_{i=1}^N$, where R_i is the point pattern for the ith word example (R restricted to some candidate interval of speech). While one can define kernels that operate directly on pairs of point patterns, in this study we applied uniform time binning to vectorize each R_i, relying on a linear duration normalization for each example.[1] If T_i is the duration of the ith example starting at time T_0, we linearly scale each $t \in R_i$ to a corresponding $t' \in R'_i$ such that $t' = (t - T_0)/T_i \in [0, 1]$. Note that attempting equivalent normalization for frame-based representations would introduce aliasing complications.

Once we have normalized all examples to unit duration, we map each R'_i to a $(M \cdot |\mathcal{P}| + 1)$-dimensional vector x_i, where M is the number of time bins (a model parameter to be tuned at validation time) and $|\mathcal{P}| = 42$ is the number of phone detectors. The first $M \cdot |\mathcal{P}|$ elements are defined such that each x_i is the supervector formed by concatenating the M-dimensional vector of binned event counts for each phone. Formally, the kth component of x_i is given by $x_i[k] = n_{j,d}$, the number of events for the jth phone in time bin d, where $j = \lfloor k/M \rfloor$ and $d = mod(k, M)$. The $(M \cdot |\mathcal{P}| + 1)$-st feature is taken to be T_i, the duration of the example. Given the training examples converted to this vector form, we can apply any standard kernel function. We limited our study to the nonlinear radial basis function (RBF) kernel, which takes the form $K(x_i, x_j) = \exp\left(-\|x_i - x_j\|^2/2\sigma^2\right)$ where σ is a kernel width parameter.

3. EXPERIMENTS

Our evaluation task used discriminative point process models to provide an alternate set of acoustic model lattice scores. We used 430

[1] While linear normalization may be sufficient for the short words we evaluate, longer, multisyllable words may benefit from kernels that operate on the point patterns directly and measure nonlinear warping between the examples.

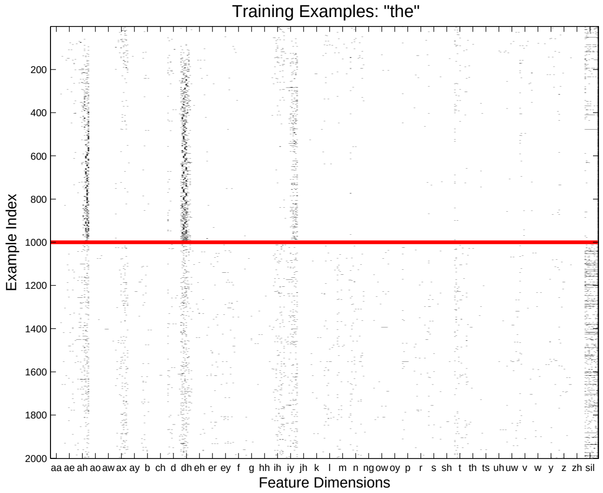

Fig. 3. Positive and negative vectorized training examples of the word "the," where counts have been clamped to 1 for viewability.

hours of Hub4 and TDT4 broadcast news data, split into 2 equal folds (denoted `fold1` and `fold2` below) for cross-validation of the various system components. We define our evaluation word set to be the following 100 most confusable words (top error producers) when the baseline recognizer described below was applied to the dev04f dataset (in order of descending number of confusions):

> the and that to is in of are it on for had he you all with there as but what we was they them or have at about will not be up their out our when way this said now like an where think then some so one if how has good could your yeah why which were two time three more me his get do day because well than no into him here going down can am after yes would world who while war very too today thing see says right really over only off my most many long

The positive examples consisted of lattice arcs labeled with the target word that were deemed correct by forced alignment with the reference transcripts. The negative examples were extracted from falsely hypothesized lattice arcs that were not consistent with the forced alignment. We evaluated the DPPM and baseline systems on two test example sets: (i) the test lattice examples alone, and (ii) the lattice examples augmented with any correct segments from the forced alignment of the test data (all positive examples) that did not make it into the lattice. Note that the extra forced alignment examples will necessarily have baseline confidences of zero, as their paths were pruned in lattice construction, and are the most difficult cases (hence the degraded performance when they are included).

3.1. Baseline Setup

We used the IBM Attila Speech Recognition Toolkit [7] to produce word lattices for our experiments and to provide the set of baseline confidence scores to which we compare our discriminative PPMs. The state-of-the-art HMM-GMM baseline acoustic model was constructed using the following advanced ASR techniques (see [7] and the references therein for details): context-dependent quinphone states built on top of 150k Gaussians, LDA, VTLN, fMLLR and fMMI speaker adaptation, MLLR, and bMMI discriminative retraining. The lattices were produced using this acoustic model and a 400M word trigram language model with the standard Attila likelihood weighting. For reference, this system produced a competitive 16.3% word error rate on the dev04f broadcast news dataset.

The lattices for `fold1` were generated by a recognizer trained on `fold2` and vice versa. Using the *acoustic model likelihoods only*, the Attila lattices for `fold2` were used to construct confusion networks (CN) [11]. This process maps each lattice arc to a confusion bin and assigns it the posterior probability of that word in the bin. Having demonstrated representative performance in prior confidence measure research [12], we used these CN posteriors as our baseline confidence scores. Note that in cases that a lattice arc was pruned in the construction of the network, it was assigned zero confidence.

3.2. Discriminative PPM Setup

Our MLP-based acoustic model was trained with a multi-stream, cascade architecture using two forms of frequency domain linear prediction coefficients as input, as described in [9, 8]. The posteriorgrams for `fold2` were produced by MLPs trained on `fold1` and vice versa. Using these posteriorgrams to define our phone detector set $\{D_p\}$ with a posterior threshold of $\delta = 0.5$, we trained regularized least squares classifiers $\{d_w\}$ for our 100 word set using the RBF kernel.[2] The first half of `fold1` was used for DPPM training, while the second half of `fold1` was held out for validation, which led to optimal model parameters of $\gamma = 1/N$ and $\sigma = 0.25$. While the individual word frequencies in each fold ranged from about 2k to 200k examples, each DPPM was trained with at most 25k randomly selected examples (all `fold2` test data was decoded regardless of quantity).[3] We used $M = 10$ time divisions (increasing resolution beyond that provided diminutive gains), resulting in 421-dimensional feature vectors, constructed according to Section 2.2.

Figure 3 displays a portion of the data used to train the classifier for the word "the," where each row represents the feature vector for a single example (the duration feature has been omitted in the image). The examples above the red line are computed from correctly hypothesized "the" lattice arcs, while those below are baseline system hallucinations. There are a few properties to note. First, we see a clear preponderance of /dh/, /iy/, and /ah/ events, arising from the correct pronunciations of the word. However, in the positive examples, the correct phonetic events are more reliably detected and are more tightly distributed in time. The negative examples display an increased presence of incorrect phonetic events, which results in the increased noise present in the bottom half of the image.

3.3. Results

Figure 4 (left) shows the word-level performance comparison between the discriminative PPM classifier scores and the CN baseline scores, where the examples were drawn from both the lattice and forced alignments. We find the equal error rates produced by the two methods to be comparable, but the correlation is weaker than might be expected. Indeed, the distinct processing chains employed by the two methods, from the front end through the final confidence scores, result in a high level of complementarity and suggest that score combination would be advantageous. Figure 4 (right) shows the word-level performance comparison between the baseline scores and the combination of DPPM and baseline scores, where we simply summed the two scores. For every word, the combined score outperforms the baseline by up to 10% absolute (50% relative).

[2] Note that support vector machines with an RBF kernel produced generally equivalent results and thus are omitted in the performance listings.

[3] For readers concerned by the whole-word training data requirement, note that two-thirds of the Hub4+TDT4 training corpus is covered by words with at least 1000 occurrences.

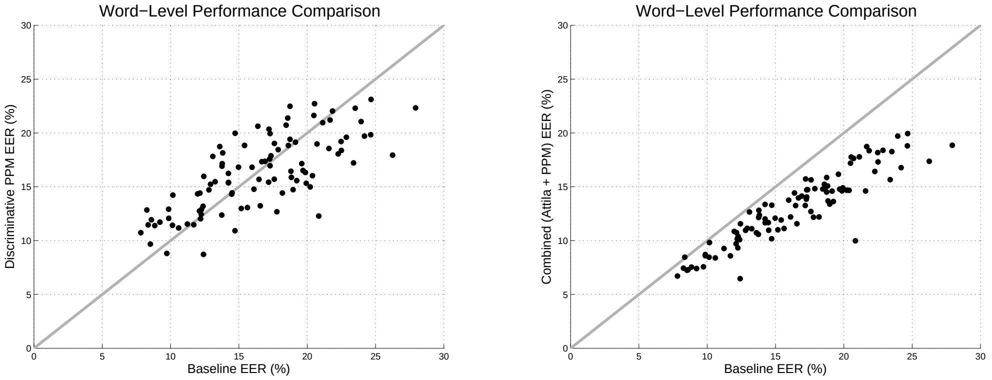

Fig. 4. Equal error rates for each word using Attila confusion network posteriors versus those of the (i) the DPPM classifier scores (left) and (ii) the combined DPPM + Attila CN scores (right).

Table 1. Average equal error rates over the 100 word set for various scores using (i) lattice examples alone and (ii) lattice examples augmented with examples from the transcript forced alignments (FA).

Scores	Lattice Only	Lattice + FA
Attila CN	12.6%	17.1%
Generative PPM	16.1%	17.8%
Discriminative PPM	14.6%	16.4%
Attila CN + DPPM	10.2%	13.4%

Finally, Table 1 lists the average EER across the word set for each score variant (including the *generative* PPM scores of [2] using the current MLP phone detectors) using both example sets. On both sets, we observe an improvement from discriminative PPM training over the original generative model. When restricted to lattice examples only, our DPPM comes remarkably close to the state-of-the-art Attila CN scores. Note that this is accomplished using solely the timing patterns of monophone events and without implementing an equivalent to the baseline feature or model adaptation. When we give our DPPM a shot at the test set forced alignment examples that the baseline system pruned, we observe an overall improvement relative to the baseline, as our model finds some of these more difficult cases manageable. System combination produces significant improvements over the baseline averages on both example sets. Note that while the interaction of the DPPM scores with a language model was not explored for the present study, we observed significant reductions in LVCSR word error rates using the SCARF framework to integrate the multiple information sources (see [8]).

4. CONCLUSIONS

We have presented a new discriminative training procedure for the point process-based acoustic modeling framework. We found that using the duration-normalized timings of sparse phone events derived from MLP-based posteriorgrams provides a suitable representation for constructing state-of-the-art word-level acoustic scores. Due to their distinct design, combining discriminative point process models scores with those produced by existing frame-based models provides a promising avenue for downstream gains.

5. ACKNOWLEDGMENTS

The author would like to thank Damianos Karakos of the Center for Language and Speech Processing (CLSP) at Johns Hopkins for his invaluable help in computing the baseline Attila lattices and confusion networks; Samuel Thomas, also of CLSP, for providing the MLP-based acoustic models; and Geoffrey Zweig of Microsoft Research for assistance in data preparation.

6. REFERENCES

[1] C.-H. Lee et al., “Word recognition using whole word and subword models,” in *Proc. of ICASSP*, 1989.

[2] A. Jansen and P. Niyogi, “Point process models for spotting keywords in continuous speech,” *IEEE Trans. Audio Speech Lang. Process.*, vol. 17, pp. 1457–1470, 2009.

[3] A. Jansen and P. Niyogi, “Detection-based speech recognition with sparse point process models,” in *Proc. of ICASSP*, 2010.

[4] D. Vergyri et al., “The SRI/OGI 2006 spoken term detection system,” in *Proc. of Interspeech*, 2007.

[5] T.J. Hazen et al., “Recognition confidence scoring and its use in speech understanding systems,” *Comp. Speech and Lang.*, vol. 16, no. 1, pp. 49–67, 2002.

[6] M. Siu and H. Gish, “Evaluation of word confidence for speech recognition systems,” *Comp. Speech and Lang.*, vol. 13, pp. 299–319, 1999.

[7] H. Soltau et al., “The IBM 2006 GALE Arabic ASR system,” in *Proc. of ICASSP*, 2007.

[8] G. Zweig et al., “Speech recognition with segmental conditional random fields: A summary of the JHU 2010 summer workshop,” in *Proc. of ICASSP*, 2011.

[9] S. Thomas, S. Ganapathy, and H. Hermansky, “Phoneme recognition using spectral envelope and modulation frequency features,” in *Proc. of ICASSP*, 2009.

[10] M. Layton and M. Gales, “Acoustic modelling using continuous rational kernels,” *J. VLSI Sig. Proc.*, vol. 48, 2007.

[11] L. Mangu, E. Brill, and A. Stolcke, “Finding consensus in speech recognition: word error minimization and other applications of confusion networks,” *Comp. Speech and Lang.*, vol. 14, pp. 373–400, 2000.

[12] D. Falavigna, R. Gretter, and G. Riccardi, “Acoustic and word lattice based algorithms for confidence scores,” in *Proc. of ICSLP*, 2002.