Constrained Kronecker Deltas for Fast Approximate Inference and Estimation

Chris Pal, Charles Sutton and Andrew McCallum
University of Massachusetts Amherst
Dept. Computer Science
Amherst, MA 01003
{pal,casutton,mccallum}@cs.umass.edu

Abstract

We are interested in constructing fast, principled algorithms for scaling up decoding and parameter estimation in probability models such as Hidden Markov Models (HMMs) and linear-chain Conditional Random Fields (CRFs) with large state spaces. In this paper we present a principled extension of beam search to beam inference and learning. We present a new approach for approximate inference based on approximating single variable potentials with a constrained, lower complexity, adaptively sized sum or mixture of Kronecker delta functions. We present inference and optimization results on synthetic and real data for HMMs and CRFs. In a number of cases, we demonstrate potential speed increases of close to an order of magnitude with little or no degradation in accuracy over exact methods for decoding and parameter estimation. We also demonstrate speed, accuracy and robustness improvements over traditional, n-best and threshold based beam methods. While we focus experiments on HMMs and CRFs, these methods should be widely applicable for inference in many factorized representations of probability distributions with large state spaces.

1 Introduction

Viterbi beam search procedures represent critically important components of Hidden Markov Model (HMM) based systems for large scale, real world speech recognition systems [2, 12, 17]. Linear-chain Conditional Random Fields (CRFs) [7] are conditional models which, when constructed for sequence labeling problems involve similar computations as in HMMs. For example, given a CRF with known parameters we are often interested in computing an analog of the "Viterbi parse" [15] which is often computed in an HMM. In both HMMs and CRFs these computations determine maximum probability configurations. When estimating the parameters of an HMM, one can use the sum-product algorithm [6] to find pairwise marginals as a step in the estimation procedure. In a CRF a similar forward-backward, sum-product pass is used repeatedly during model optimization to compute expectations [7]. These computations are used to compute the gradient of the log likelihood, have quadratic time complexity and are thus is very expensive when state spaces are large. We are interested in speeding up inference so as to solve decoding and optimization problems in HMMs and CRFs with large state spaces.

Recently, a number of algorithms have been proposed which involve incrementally computing approximations to probability distributions using "local" updates of random variables in a graphical structure. Expectation Propagation [8] computes an approximation to complex posteriors over random variables through matching the expectations of moments. Other recent related work on expectation consistent free energies [11] has employed optimization methods which minimize free energies subject to expectation constraints. In these approaches free energies are defined by the KL divergence between an approximate, tractable distribution q and an intractable distribution p and subject the approximate distributions to constraints.

In the context of Bayesian networks and inference in Expert Systems the work in [5] investigated the effect of truncating small numbers and demonstrated significant gains in speed. Other more recent work has also looked at approximations in terms of "message errors" [3] in belief propagation. It has also been shown that sum product messages of belief propagation [6] correspond to the fixed point equations of the Lagrangian formed from a free energy of an MRF when it is minimized subject to marginalization and normalization constraints [18]. This analysis has led to some further

understanding of "loopy belief propagation" schemes in probability models with cycles. Variational Methods have also been used to construct approximate distributions consisting of structured or factorized components [4]. Variational message passing algorithms extend "mean field" methods and can be constructed such that one computes updates to clusters of factorized approximations which involve only localized variables in graphical models [16].

In this paper we extend beam search to the notion of beam inference and cast the procedure as a variational inference problem. Rather than constraining expectations, constraining graphical structures or factorizations of distributions, our approximations presented here involve constructing constrained, lower complexity distributions formulated in terms of Kronecker deltas. Our constraints are based on finding a minimal complexity approximate distribution with Kullback-Leibler (KL) divergence from a full "beam" within some ϵ and a minimum beam size constraint. We show robustness and efficiency improvements over simple beam search methods and present CRF optimization results in which models are obtained in less than 20% of the time required using exact forward backward methods for learning with no loss in test set accuracy. Using our method we are now able to train CRFs with large state spaces in about 6 hours whereas previously training took over a day.

In the following sections we review message passing algorithms to compute marginals and maximum probability configurations, we briefly review HMMs and CRFs and discuss current methods for learning CRFs. We then present a derivations of our new algorithms based on constraining potentials to have the form of a sum of Kronecker deltas. We then present decoding and learning results.

2 Inference and Message Passing

Message passing algorithms can be used to compute both marginals and maximum probability configurations in probability models. Here, we briefly review the sum-product and max-product algorithms.

2.1 Computing Marginals

Marginal distributions of variables in tree structured graphical models can be efficiently computed using the sum-product algorithm. The sum product algorithm for Factor Graphs [6] can be written in terms of two types of messages. These messages take the form of variable to function messages, defined as:

$$\mu_{v_i \to f_j}(v_i) = \prod_{f_k \in n(v_i) \setminus f_j} \mu_{f_k \to v_i}(v_i) \tag{1}$$

and function to variable messages are defined as:

$$\mu_{f_j \to v_i}(v_i) = \sum_{v_k \in n(f_j) \setminus v_i} \left(f_j(n(f_j)) \prod_{v_k \in n(f_j) \setminus v_i} \mu_{v_k \to f_j}(v_k) \right) \tag{2}$$

We can think of the messages from variable i to function j as local approximations to single variable marginals where we have excluded the impact of function j and the rest of the graph connected through function j. We can write this approximate marginal as

$$\begin{aligned} \hat{p}_{i,j}(v_i) &\equiv \frac{\mu_{v_i \to f_k}(v_i)}{\sum_{v_i} \mu_{v_i \to f_k}(v_i)} \\ &= \frac{1}{Z} \mu_{v_i \to f_k}(v_i) \end{aligned} \tag{3}$$

2.2 Computing Max-Configurations

An attractive aspect of the sum-product algorithm for computing marginals in a tree structured network is that a simple modification of the algorithm can be used to compute the maximum probability configuration of the joint assignment of variables in the model. The function to variable messages of (2) in the message passing algorithm of Section (2.1) can be modified such that the sum is replaced by a maximum over the same variables as the sum. The update can thus be written:

$$\mu_{f_j \to v_i}(v_i) = \max_{v_k \in n(f_j) \setminus v_i} \left(f_j(n(f_j)) \prod_{v_k \in n(f_j) \setminus v_i} \mu_{v_k \to f_j}(v_k) \right) \tag{4}$$

3 HMMs, CRFs and Inference

In the work here, we are particularly interested in inference in chain structured probability models. Thus, we briefly review HMMs and CRFs. We also briefly review the current techniques for CRF parameter estimation.

3.1 HMMs

HMMs are a classical type of generative sequence model. Define an observation sequence of *random categorical variables* as $\mathbf{x} = x_1, \ldots, x_n$ and a sequence of random categorical variables for the "label" variables as $\mathbf{y} = y_1, \ldots, y_n$.

$$p(\mathbf{y}, \mathbf{x}) = \prod_{i=1}^{n} p(x_i|y_i) p(y_i|y_{i-1}) \tag{5}$$

where for simplicity we define $p(y_1|y_0) = p(y_1)$. For inference and estimation we are often interested in computing posterior marginal distributions over hidden labels. For many decoding applications, we are interested in quickly computing the most probable $\mathbf{y}$.

3.2 CRFs

A Conditional Random Field (CRF) represents a conditional probability distribution using a set of features. Define subsets of our random variables of sequences as $\mathrm{x} \subset \mathbf{x}$ and $\mathrm{y} \subset \mathbf{y}$. For example y_q and x_q might specify the subsets $\mathrm{y}_q = \{x_2, x_3\}$, $\mathrm{x}_q = \{x_1, x_2, x_3\}$. A CRF represents the conditional probability of a label sequence given an observation sequence as

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{q \in Q} \Psi_q(\mathrm{y}_q, \mathrm{x}_q), \tag{6}$$

where

$$Z(\mathbf{x}) = \sum_{\mathbf{y}} \prod_{q \in Q} \Psi_q(\mathrm{y}_q, \mathrm{x}_q) \tag{7}$$

is a normalizing factor over *all* state sequences or all states of $\mathbf{y}$ for each possible state sequence sequence or configuration of $\mathbf{x}$. In a CRF there are typically additional factorizations of the potentials Ψ_q using k feature functions $\{f_k\}$ for each q such that

$$\Psi_q(\mathrm{y}_q, \mathrm{x}_q) = \exp\left(\sum_k \lambda_k f_k(\mathrm{y}_q, \mathrm{x}_q) \right), \tag{8}$$

where λ_k are the parameters or feature weights for the model.

3.3 Inference and Optimization in CRFs

In a CRF, when we observe the values of an observation or label sequence, we have an *instantiation* of random variables, which we denoted by $\tilde{\mathbf{x}}$ or $\tilde{\mathbf{y}}$ respectively. Given an instantiation of an observation sequence $\tilde{\mathbf{x}}$ we can represent the conditional distribution $p(\mathbf{y}|\tilde{\mathbf{x}})$ using an undirected graphical structure on the potentials associated with the cliques $q \in Q$ defined by the dependency structure of the unobserved random variables $\mathbf{y}$. One can thus illustrate this conditional distribution using a factor graph where the observations $\tilde{\mathbf{x}}$ are "absorbed" into the potential functions of the cliques of (8). The computation of $Z(\tilde{\mathbf{x}})$ in (7) also simplifies to the computation of a scalar quantity. "Decoding" or finding the most probable $\mathbf{y}$ in a tree or chain structured CRF can thus be computed using the max-product messages of Section 2.2 with potential functions defined by (8).

It is possible to optimize the parameters of a CRF by seeking the minimum of the gradient of the log-likelihood of the CRF given a data set of observation and label sequences, $T = \{\tilde{\mathbf{x}}_i, \tilde{\mathbf{y}}_i\}$ for $i = 1, \ldots, N$. If we define $\boldsymbol{\lambda}$ as a parameter vector, F as a feature function and $\mathbf{F}(\mathbf{y}, \mathbf{x}) = \sum_q F(\mathrm{y}_q, \mathrm{x}_q)$ as the *global feature function*, then the gradient of the log-likelihood $\mathcal{L}$ with respect to the model parameters is given by

$$\begin{aligned} \nabla_{\boldsymbol{\lambda}} \mathcal{L} &= \nabla_{\boldsymbol{\lambda}} \left(\sum_i \log p(\tilde{\mathbf{y}}_i | \tilde{\mathbf{x}}_i, \boldsymbol{\lambda}) \right) \\ &= \sum_i \left(\mathbf{F}(\tilde{\mathbf{y}}_i, \tilde{\mathbf{x}}_i) - E_p \langle \mathbf{F}(\mathbf{y}_i, \tilde{\mathbf{x}}_i) \rangle \right), \end{aligned} \tag{9}$$

where $E_p\langle \cdot \rangle$ denotes the expectation under the distribution $p = p(\mathbf{y}_i | \tilde{\mathbf{x}}_i, \tilde{\boldsymbol{\lambda}})$. For a chain structured CRF this expectation can then be expressed as

$$\begin{aligned} E_p \langle \mathbf{F}(\mathbf{y}_i, \tilde{\mathbf{x}}_i) \rangle &= \sum_{l=1}^{L-1} p(\mathrm{y}_l | \tilde{\mathbf{x}}, \tilde{\boldsymbol{\lambda}}) \log F(\mathrm{y}_l, \tilde{\mathbf{x}}, \tilde{\boldsymbol{\lambda}}) \\ &= Z(\tilde{\mathbf{x}}, \tilde{\boldsymbol{\lambda}})^{-1} \sum_l \boldsymbol{\alpha}_l \Phi_l \boldsymbol{\beta}_l^T, \end{aligned} \tag{10}$$

where $\Phi_l = \log F(\mathrm{y}_l, \tilde{\mathbf{x}}, \tilde{\boldsymbol{\lambda}})$ and $\mathrm{y}_l = \{y^{l-1}, y^l\}$ i.e. pairwise cliques and $\boldsymbol{\alpha}_l$ and $\boldsymbol{\beta}_l$ are the variable to function messages from (1). It is important to observe that (1) thus requires forward backward inference once for each sequence, per iteration. For data sets with large state spaces, this procedure can take days of computation.

In practice, we use the approach in [14] to perform gradient based optimization of CRFs. As such, (10) and the messages of (1) are embedded within a limited memory, quasi-Newton (L-BFGS) [10] optimizer. In Section 4 we will construct algorithms to compute approximate conditional marginals so as run significantly faster than the standard, exact forward backward algorithm but minimize the impact of the approximation. Further, we shall define this criterion more formally in terms of KL divergences.

4 Inference and Learning with Constrained Kronecker Deltas

An intuitive way to think of our goals in this work is in terms of deriving efficient, principled method for determining which states to neglect in max-propagation and sum-propagation message passing schemes. Recall, we are interested in these goals so as to dramatically improve the speed of inference for both decoding and estimation tasks within optimization or "learning" algorithms. To achieve these goals and derive such algorithms, in the following sub-sections we derive the following:

- The Kronecker delta distribution with N fixed non-zero elements which minimizes $KL(q||p)$ for an arbitrary discrete distribution p.
- For any number of non-zero states N, the Kronecker delta distribution which minimizes $KL(q||p)$.

- An efficient method to find the lowest complexity q such that $KL(q||p) \leq \epsilon$

Using these results we then present new approximate algorithms for asynchronous and beam versions of sum and max propagation based on constraining Kronecker delta potentials for variables with large state spaces.

4.1 Kronecker Deltas and KL Divergences

Consider computing the KL Divergence $KL(q||p)$ between a discrete probability distribution p and q where p is some arbitrary discrete probability distribution and q is a weighted sum of Kronecker deltas placing probability mass over some subset of the state space. If x is a discrete state index $x = \{1, \ldots, n\}$ we will write q as

$$q(x) = \sum_{i \in I} q_i \delta_i(x), \tag{11}$$

where $I = \{1, \ldots, k\}$, i.e. a set of indices for the non-zero states and $\delta_i(x_j) = \delta_{ij} = 1$ for $j = i$. We also have the additional constraint that

$$\sum_{i \in I} q_i = 1 \tag{12}$$

Now, given distribution p and approximate Kronecker distribution q with some *arbitrary* indices I, consider the task of minimizing the KL divergence between q and p subject to a normalization constraint. We can express this as the task of minimizing a Lagrangian of the form

$$\sum_{i \in I} q_i \log q_i - \sum_{i \in I} q_i \log p_i + \lambda \left(\sum_{i \in I} q_i - 1 \right) \tag{13}$$

Differentiating this Lagrangian with respect to q_i allows us to determine the relationship

$$q_i = \exp(\log p_i - 1 - \lambda). \tag{14}$$

Then, substituting this back into the normalization constraint we can derive that

$$\lambda = \log \left(\sum_{i \in I} p_i \right) - 1 \tag{15}$$

and the somewhat intuitive result that

$$q_i = \frac{p_i}{\sum_{i \in I} p_i} \tag{16}$$

Thus, we can write our approximating distribution as the sum of weighted, normalized Kronecker deltas

$$q(x) = \sum_{i \in I} q_i \delta_i(x) \tag{17}$$

4.2 Minimizing the KL divergence

We are interested in the optimal set of indices I which minimize $KL(q||p)$. We would like to show that the divergence is minimized when $I = \{i_1, \ldots, i_k\}$ consists of the indices of the largest k values of the discrete distribution p. First, define

$$Z(I) = \sum_{i \in I} p_i \tag{18}$$

then Z^* and I^* defined as

$$Z^* = \max_I Z(I), \quad I^* = \arg\max_I Z(I), \tag{19}$$

are clearly given by indices of the top k values of p. Given these definitions we can thus derive that

$$\begin{aligned} &\arg\min_q KL(q||p) \\ &\quad = \arg\min_I \left\{ \arg\min_{\{q_i\}} \sum_x q(x) \log \frac{q(x)}{p(x)} \right\} \\ &\quad = \arg\min_I \left\{ \arg\min_{\{q_i\}} \sum_{i \in I} q_i \log \frac{q_i}{p_i} \right\} \end{aligned} \tag{20}$$

which we know from (16) in section 4.1

$$\begin{aligned} &= \arg\min_I \left\{ \sum_{i \in I} \frac{p_i}{Z(I)} \log \frac{p_i / Z(I)}{p_i} \right\} \\ &= \arg\min_I \left\{ \frac{1}{Z(I)} \sum_{i \in I} p_i \log \frac{1}{Z(I)} \right\} \\ &= \arg\max_I \{ \log Z(I) \} \\ &= I^* \end{aligned} \tag{21}$$

4.3 The Lowest Complexity q with $KL(q||p) \leq \epsilon$

We wish to find the minimal complexity q with $KL(q||p) \leq \epsilon$. From Section 4.2 we know we that the minimum divergence of q for any desired number of non-zero entries can be encoded by sorting the elements p_i and representing the first j indices by the set I^j. If we let $Z_j = Z(I^j)$ be the partition function for each distribution in the set of distributions $\mathcal{Q} = \{q^1, q^2, \ldots, q^N\}$, then using the result in section 4.2 we know that the KL divergence between p and distribution q^j with minimal divergence for each $j = 1 \ldots N$ is given by

$$KL(q^j||p) = -\log Z(I^j). \tag{22}$$

Since we can easily compute Z_{j+1} from Z_j, we perform a single pass across the sorted state space of p and easily find the minimal complexity q such that $KL(q||p) \leq \epsilon$. We write the set of indices satisfying this property as

$$I^*_{KL \leq \epsilon} = I^j \mid KL(q^j||p) \leq \epsilon \tag{23}$$

4.4 New Constrained Beam Algorithms

One can use our approach of finding the minimal complexity q with $KL(q||p) \leq \epsilon$ in the context of "loopy" [9] local message passing algorithms and in the context of approximate "forward backward" and "sum-product" algorithms consisting of a single forward beam pass followed by a single backward beam message passing scheme. In the case of loopy updates we have $q(v_i)$ approximating $\hat{p}(v_i)$ which can be written

$$q(v_i) \approx \hat{p}(v_i) = \frac{1}{Z} \prod_{f_k \in n(v_i)} \mu_{f_k \to v_i}(v_i) \qquad (24)$$

In the constrained beam algorithms investigated in this paper we are interested in chain structure models with large state spaces. The algorithms presented here generalize easily to tree structured graphs. For learning we compute approximate "forward" marginal probabilities $\hat{p}_f(v_i) = \hat{\alpha}_i$ sequentially from (24) in a forward pass such that $n(v_i)$ are restricted to the sequential predecessors in the chain. In the backward pass $\hat{p}_b(v_i) = \hat{\beta}_i$ we use the neighbors $n(v_i)$ which are successors. In the forward pass we use (24) and (23) to determine the non-zero element indices $I^*_{i|KL \leq \epsilon}$ then eliminate or "zero out" states effectively constructing our beam for $\hat{p}_f(v_i) = \hat{\alpha}_i$. The backward pass of our algorithm operates only upon states in the forward beam defined by $I^*_{i|KL \leq \epsilon}$.

In the case of approximate, beam "Viterbi decoding" or "max-product" updates, we can similarly define loopy variants of our algorithm for which updates of the max marginal $\hat{\phi}(v_i)$, analogous to $\hat{p}(v_i)$ can be written using $\psi(v_i) \approx$

$$\hat{\phi}(v_i) = \prod_{f_k \in n(v_i)} \max_{v_k \in n(f_j) \setminus v_i} \left(f_j(n(f_j)) \prod_{v_k \in n(f_j) \setminus v_i} \mu_{v_k \to f_j}(v_k) \right) \qquad (25)$$

where we compute a local Z and apply (24) to determine (23). In the results in Section 5 we use a fixed beam sized version of the loopy updates (25) as a comparison algorithm. However, most experiments are performed with a one pass, max forward beam with potentials $\psi_f(v_i)$ and I^*_i.

5 Results and Analysis

Here we present results for inference in the context of Viterbi decoding for synthetic data for HMMs and CRFs. We also present results from the NetTalk dataset for decoding in CRFs. We present inference results in the context of learning for parameter estimation in CRFs using synthetic data and the NetTalk dataset. Our initial experiments and the results presented here have also lead us to introduce an additional minimum beam size constraint when determining both sum and max beam indices I^*_i such that $KL \leq \epsilon$ and $|I_i| \geq N$.

5.1 Decoding Experiments

For our decoding experiments we generated synthetic data from an HMM for sequences of length 75. Transition matrix entries were sampled from a Dirichlet with $\alpha = .1$ and emission matrices were generated from a mixture of a low entropy, sparse conditional distribution with 10 non-zero elements and a high entropy Dirichlet with $\alpha = 10^4$, with priors of .75 and .25 respectively. The goal was to simulate mostly highly informative states and some less informative states. In the following experiments of Tables 1, 2 and 3, to examine the behavior of traditional beams and our new approach, we have performed recognition accuracy tests using an HMM with the parameters we used to generate the data.

In Table 1 we present results where we implemented beam search in an HMM based on a threshold and only expanding states with a current log score within the factor shown of the best log score. In Table 2 we present results using fixed sized beams of varying sizes. In Table 3 we present results consisting of our $KL \leq \epsilon$, $|I_i| \geq N$ algorithm and a simple backward corrective beam algorithm.

Our simple backward beam algorithm consists of exploring at each variable the single state from the known forward Viterbi path and the single state with the best log score in the backward path which was not included in the forward path. The backward beam algorithm is also constrained to always explore any states which have a log score that is better than the known forward Viterbi path, a quantity easily determined by the forward and backward log scores.

	Beam Thresholds				
	1.05	1.1	1.25	1.5	2
Accuracy	85.8	90.1	93.0	**95.4**	95.4
Avg. Beam Size	10.3	17.1	26.2	**30.4**	32.2

Table 1: *Threshold Viterbi Beam Search in an HMM*: Recognition accuracy using a threshold based on a factor of the best paths log score for a given time step. Recognition accuracy is shown for synthetic data generated as described above. Exact Viterbi accuracy is 95.4%.

Analysis of these results reveals that the thresholds in Table 1 can achieve the exact Viterbi accuracy of

	Fixed Beam Sizes					
	2	3	4	5	6	7
Accuracy	65.8	76.0	83.5	86.5	89.0	90.6
	8	9	10	15	20	**25**
Accuracy	91.6	92.3	93.3	94.2	95.2	**95.4**

Table 2: *N-best Viterbi Beam Search in an HMM*: Recognition accuracy on synthetic data for an N-best Viterbi Beam Search. Exact Viterbi accuracy is 95.4%.

	Beam / Constraint Size				
	2	3	4	5	6
Std. N-best	65.8	76.0	83.5	86.5	89.0
Bkwd. Corr.	86.1	87.7	91.9	93.9	94.1
$KL \leq \epsilon$	94.2	94.8	**95.4**	95.4	95.4
	Average Beam Size				
Bkwd. Corr.	9.4	6.1	4.2	4.4	2.7
$KL \leq \epsilon$	8.8	8.9	**9.6**	10.3	11.1

Table 3: *Comparing New Algorithms with N-best Viterbi Beam Search in an HMM*: Recognition accuracy on synthetic data for an n-best Forward Viterbi Beam Search, the n-best Forward Viterbi Beam with a backward corrective beam, and our adaptive forward beam based on $KL \leq \epsilon = .001$ and a minimum beam size constraint. Exact Viterbi accuracy is 95.4%.

95.4% but explore an average of 30.4 states per variable. Table 2 illustrates how a fixed beam of size between 20 and 25 could also produce the the exact Viterbi accuracy. While Table 3 shows that the exact Viterbi accuracy can be achieved while exploring an average of only 9.6 states using a minimal complexity $KL \leq .001$ criterion with an additional $|I_i| \geq 4$ constraint. Consider again the results in Table 1. To compare with our new algorithm, we introduced an additional minimum beam size constraint of 4 and 10 respectively for the beams obtained with threshold 1.05. These experiments resulted in an accuracy of 91.9% and 94.5% with an average beam size of 9.9 and 13.5 respectively.

In the experiments of Tables 4 and 5 we generated synthetic data from an HMM with 100 hidden states and 100 observed states. The model we used here had sparse transition and emission matrices such that there were 5 transitions per state and 5 emission values per states. We used 50 sequences of length 75 for optimizing the model and 50 examples for testing the model. Using standard Viterbi decoding, the CRF for this data had a recognition accuracy of 87.3%. Table 4 present the beam results based on thresholds. In Table 5 we compare the forward, fixed beam results with:

	Beam Thresholds					
	.99	.98	.97	.95	.9	.5
Accuracy	77.4	83.2	84.7	85.9	87.1	**87.3**
Avg. Size	7.3	37.7	57.6	74.3	85.7	**95.2**

Table 4: *Threshold Viterbi Beam Search in a CRF*: Recognition accuracy using a threshold based on a factor of the best paths log score for a given time step. Recognition accuracy is shown for synthetic data generated as described above. Exact Viterbi accuracy is 87.3%.

	Beam / Constraint Size				
	3	5	10	15	20
Std. N-best	57.8	72.4	82.3	85.4	86.2
Loopy Max-P	59.1	74.1	84.3	86.3	86.6
Bkwd. Corr.	62.4	73.1	83.2	85.8	86.4
$KL \leq .5$	79.7	80.5	83.2	85.8	86.3
$KL \leq .15$	**87.2**	87.2	86.9	87.1	87.1
	Average Beam Size				
$KL \leq .5$	4.9	6.2	10.2	15.0	20.0
$KL \leq .15$	**22.3**	22.6	24.0	25.9	28.3

Table 5: Comparing CRF recognition accuracy on synthetic HMM data for Viterbi Beam Search, our constrained max field algorithm in a CRF. Exact Viterbi decoding had an accuracy of 87.3%.

1. A single backward pass of the loopy max product updates of (25) in Section 4.4 with the beam width fixed.

2. Our backward corrective beam algorithm.

3. Our $KL \leq \epsilon$, $|I_i| \geq N$ adaptive forward beam with a relatively low and a relatively high ϵ for the model.

We have also optimized a CRF for the NetTalk data [13] and performed decoding experiments. The CRF we constructed had 52 states, and was optimized using 19075 examples and tested using 934 examples. Standard Viterbi parses of the optimized CRF produced an accuracy of 91.6%. Table 6 summarizes accuracy results for fixed beam sizes. Our other experiments found that both the $KL < \epsilon$ method and the threshold methods produced the exact Viterbi accuracy when the average number of states explored was 14.

5.2 Learning Experiments

For our learning experiments we have optimized CRFs using synthetic HMM data generated from a 100 state HMM with the same parameters as the CRF decoding

	Beam / Constraint Size					
	2	3	4	5	6	12
Accuracy	82.7	88.3	90.4	91.2	91.4	91.6

Table 6: Comparing recognition accuracy on the NetTalk data set using a CRF with a Viterbi accuracy of 91.6%.

experiments. Again, we use 50 sequences for optimization and 50 sequences for testing accuracy. In all cases we use exact Viterbi decoding for to compute our accuracy. Figure 1 illustrates learning curves comparing our $KL \leq \epsilon$, $|I_i| \geq N$, forward backward algorithm for $\epsilon = .5, N = 30$ with an fixed beam with an average size the same as the average number of states in the KL beam and the exact forward backward optimization.

From Figure 1 we see that our L-BFGS optimizer terminates in less than a quarter of the time with the beam methods. The fixed beam method reaches its best log likelihood in approximately half the time of the exact forward backward optimizer and our constrained KL beam algorithm reaches its convergence in roughly on third of the time. Further, our KL beam also achieves the same accuracy on our test set as the exact algorithm.

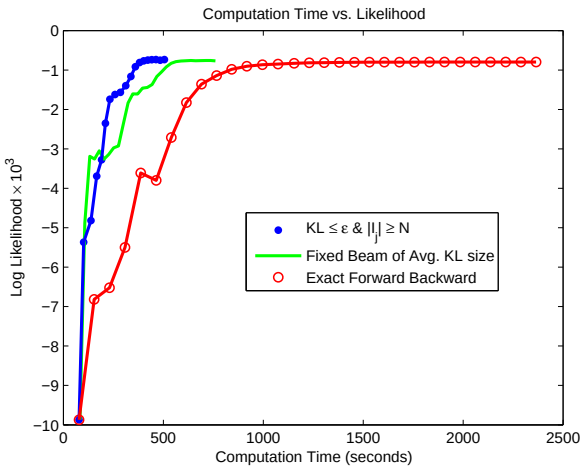

Figure 1: Accuracy for: 1. $KL \leq \epsilon$, (88.5) 2. Fixed Beam of Avg. KL Size, (87.8) 3. Exact Forward Backward (88.5)

Figure 2 shows a re-scaled version of Figure 1 but also adds the results of training with a fixed beam of the minimum beam size for the KL beam and a threshold based beam which explores on average roughly the same number of states as the KL beam. In the case of the smaller, fixed beam of size N, our L-BFGS optimizer terminated with an error as a result of the noisy gradient computation. In the case of the threshold beam, the gradients of the optimization were erratic, but L-BRFS did terminate normally. However the recognition accuracy of the final model was low, at 67.1%.

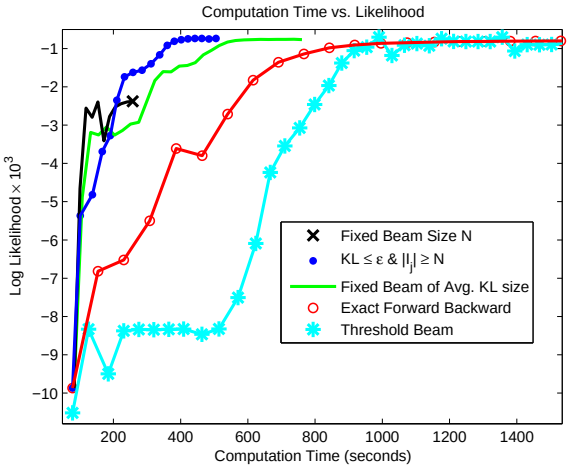

Figure 2: A re-scaled version of the curves from Figure 1 with the introduction of a fixed beam of size N optimization and a threshold beam which explored an average of 45 states.

Finally, in Figure 3 we present run time, model likelihood and accuracy results for the 52 state CRF optimized using the NetTalk data with training and testing partitions as described in Section 5.1. Our optimization routine used a fast initialization procedure in which only 12% of the data was used to initialize CRF parameters. We used the beam methods during the complete optimization run and during this initialization period. During these subset initializations, our experiments with a threshold beam set such that it explored an average of 36 states produced initial parameter estimates which had a test set accuracy of 67%. Our KL method, a fixed size beam of average KL size and exact forward backward all had accuracies of 74%. Further, during the complete run, the threshold beam gradient estimates were so noisy that our L-BFGS optimizer was unable to take a complete step. In the experiments of Figure 3, $\epsilon = .005$ and $N = 10$. Exact forward backward training produced a test set accuracy of 91.6%. In these experiments fixed beam optimization using the average size of our KL beam ($N = 20$) terminated normally but very noisy intermediate gradients were found in the terminating iteration. The result was a much lower accuracy of 85.7%. In contrast, our KL beam achieved an accuracy of 91.7% in a less than 20% of the time it took to

optimize the CRF using exact forward backward.

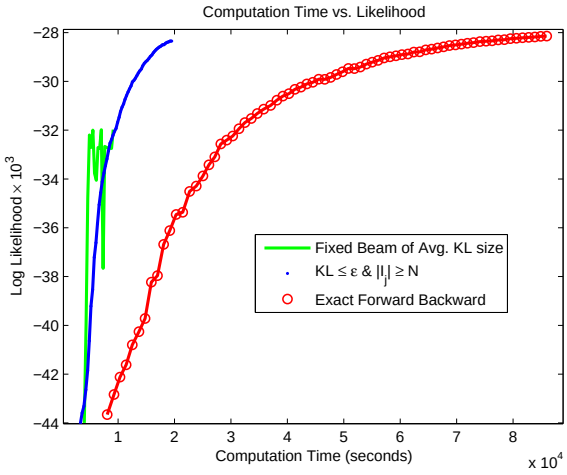

Figure 3: Comparing Different Forward Backward Beam Learning Algorithms for NetTalk. Accuracy for: 1. Fixed Beam of Avg. KL Size, (85.7) 2. $KL \leq \epsilon$, (91.7) 3. Exact Forward Backward (91.6)

6 Conclusions

We have presented a principled method for significantly speeding up decoding and learning tasks in HMMs and CRFs. We also have presented experimental work illustrating the utility of our approach. As future work, we believe a promising avenue of exploration would be to explore adaptive strategies involving interaction of our L-BFGS optimizer, detecting excessively noisy gradients and automatically setting ϵ values. While the results we have presented here were applied to experiments with HMMs and chain structured CRFs, we believe this approach should be more generally applicable.

References

[1] R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer, 1999.

[2] X. Huang, A. Acero, and H. W. Hon. *Spoken Language Processing: A Guide to Theory Algorithms and System Development*, chapter 12. Prentice Hall, New Jersey, 2001.

[3] Alexander T. Ihler, John W. Fisher, and Alan S. Willsky. Message errors in belief propagation. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA, 2005.

[4] T. Jaakkola. *Variational Methods for Inference and Estimation in Graphical Models*. PhD thesis, MIT, 1997.

[5] F. Jensen and S. K. Andersen. Approximations in bayesian belief universes for knowledge-based systems. *Proceedings of the 6th Conference on Uncertainty in Artifcial Intelligence, Cambridge, Mass.*, pages 162–169, 1990.

[6] F.R. Kschischang, B.J. Frey, and H.A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Information Theory*, 47(2):498–519, February 2001.

[7] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001.

[8] Thomas Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, MIT, 2001.

[9] Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *The proceedings of UAI, Uncertainty in Artificial Intelligence*, pages 467–475, 1999.

[10] J. Norcedal and S. J. Wright. *Numerical Optimization*. Springer, 1999.

[11] Manfred Opper and Ole Winther. Expectation consistent free energies for approximate inference. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA, 2005.

[12] Mosur K. Ravishankar. *Effcient Algorithms for Speech Recognition*. PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1996.

[13] T.J. Sejnowski and C.R. Rosenberg. Nettalk: a parallel network that learns to read aloud. *Cognitive Science*, 14:179–211, 1990.

[14] Fei Sha and Fernando Pereira. Shallow parsing with conditional random fields. In *Proceedings of Human Language Technology-NAACL 2003*, Edmonton, Canada.

[15] A.J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–269, April 1967.

[16] J. Winn. *Variational Message Passing and its Applications*. PhD thesis, University of Cambridge, 2004.

[17] Monika Woszczyna. *Fast Speaker Independent Large Vocabulary Continuous Speech Recognition*. PhD thesis, Universität Karlsruhe, 1998.

[18] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Understanding belief propagation and its generalizations. pages 239–269, 2003.