SINGLE-CHANNEL SPEECH SEPARATION AND RECOGNITION USING LOOPY BELIEF PROPAGATION

Steven J. Rennie, John R. Hershey, Peder A. Olsen

IBM T.J. Watson Research Center

(sjrennie, jrhershe, pederao)@us.ibm.com

ABSTRACT

We address the problem of single-channel speech separation and recognition using loopy belief propagation in a way that enables efficient inference for an arbitrary number of speech sources. The graphical model consists of a set of N Markov chains, each of which represents a language model or grammar for a given speaker. A Gaussian mixture model with shared states is used to model the hidden acoustic signal for each grammar state of each source. The combination of sources is modeled in the log spectrum domain using non-linear interaction functions. Previously, temporal inference in such a model has been performed using an N–dimensional Viterbi algorithm that scales exponentially with the number of sources. In this paper, we describe a loopy message passing algorithm that scales linearly with language model size. The algorithm achieves human levels of performance, and is an order of magnitude faster than competitive systems for two speakers.

Index Terms— Speech separation, loopy belief propagation, factorial hidden Markov models, ASR, Iroquois, Algonquin, Max model.

1. INTRODUCTION

Existing automatic speech recognition (ASR) research has focused on single-talker recognition. In many scenarios, however, the acoustic background is complex, and can include speech from other talkers. Such input is easily parsed by the human auditory system, but is highly detrimental to the performance of conventional ASR systems. The recently introduced Pascal Speech Separation Challenge (SSC) involves recognizing a target speaker in the presence of a simultaneously speaking masker, using a single channel (see [1] for details, and a review of the state of the art).

The system presented in [2] is currently the best-performing system on the SSC, and outperforms human listening results on the task. The performance of this system hinges on the efficacy of the separation component of the system, which models each speaker by a layered, factorial hidden Markov model (HMM). In [2], approximations were used to make inference in this model tractable, but inference still scaled exponentially with the number of sources. When the speaker vocabulary is large or there are more than two talkers, we have to find more efficient methods.

Loopy belief propagation (LBP) has in recent years been successfully applied in many fields—including communications, computer vision, and molecular biology—to solve inference problems that are intractable using exact methods [3, 4].Despite the prominent use of belief propagation algorithms in ASR research and commercial applications (such as the Viterbi algorithm and the forward-backward algorithm for HMMs), and the importance of computational efficiency, little work has investigated using LBP for ASR [5].

In this paper, we present a loopy belief propagation algorithm for multi-talker speech separation and recognition using a single channel. The algorithm outperforms human listeners on the SSC task, at a fraction of the computational cost of previously published systems that can achieve such performance.

2. SPEECH SEPARATION MODELS

We use the same two-speaker model detailed in [2], and depicted in Figure 2(a). The model consists of an *acoustic model* and a *temporal dynamics model* for each speaker (see Figure 1), as well as a *interaction model*, which describes how the source features are combined to produce the observed mixture spectrum. We also use all of the optimizations given in [2] when doing exact inference in this model.

Acoustic Model: For a given speaker, a, we model the conditional probability of the log-power spectrum of each source signal $\mathbf{x}^a$ given a discrete acoustic state s^a as Gaussian, $p(\mathbf{x}^a|s^a) = \mathcal{N}(\mathbf{x}^a; \boldsymbol{\mu}_{s^a}, \boldsymbol{\Sigma}_{s^a})$, with mean $\boldsymbol{\mu}_{s^a}$, and covariance matrix $\boldsymbol{\Sigma}_{s^a}$. For efficiency and tractability we restrict the covariance to be diagonal. This means that $p(\mathbf{x}^a|s^a) = \prod_f \mathcal{N}(x^a_f; \mu_{f,s^a}, \sigma^2_{f,s^a})$, for frequency f. Hereafter we drop the f when it is clear from context that we are referring to a single frequency. In this paper we use $D_s = 256$ gaussians to model the acoustic space of each speaker.

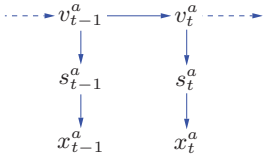


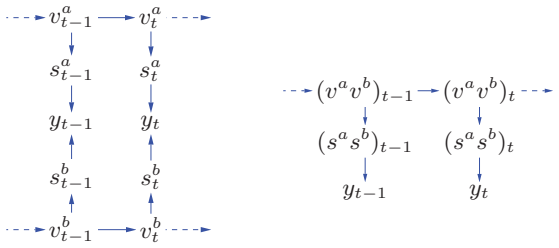
Fig. 1. Generative model for the features, x^a, of single source: an HMM with grammar states, v^a, sharing common acoustic states, s^a.

Grammars: The task grammar is represented by a sparse matrix of state transition probabilities, $p(v^a_t|v^a_{t-1})$. The association between the grammar state v^a and the acoustic state s^a is captured by the transition probability $p(s^a|v^a)$, for speaker a. These are learned from clean training data using inferred acoustic and grammar state sequences.

3. SPEECH INTERACTION MODELS

The short-time log spectrum of the mixture y_t, in a given frequency band, is related to that of the two sources x^a_t and x^b_t via the *interaction model* given by the conditional probability distribution, $p(y_t|x^a_t, x^b_t)$. The joint distribution of the observation and source features in one feature dimension, given the source states, is:

$$p(y_t, x^a_t, x^b_t|s^a_t, s^b_t) = p(y_t|x^a_t, x^b_t)p(x^a_t|s^a_t)p(x^b_t|s^b_t). \quad (1)$$



(a) Two speaker model (b) Product Model

Fig. 2. a) Generative model of mixed features. The source models are combined with an interaction model to explain the data. Here x^a and x^b have been integrated out. b) The same model with combined acoustic and grammar states to eliminate loops.

To infer and reconstruct speech we need to compute the likelihood of the observed mixture given the acoustic states,

$$p(y_t|s_t^a, s_t^b) = \int p(y_t, x_t^a, x_t^b|s_t^a, s_t^b)\mathrm{d}x_t^a\ \mathrm{d}x_t^b, \tag{2}$$

and the posterior expected values of the sources given the acoustic states and the observed mixture,

$$E(x_t^a|y_t, s_t^a, s_t^b) = \int x_t^a p(x_t^a, x_t^b|y_t, s_t^a, s_t^b)\ \mathrm{d}x_t^a\ \mathrm{d}x_t^b, \tag{3}$$

and similarly for x_t^b. These quantities, combined with a prior model for the joint state sequences $\{s_{1..T}^a, s_{1..T}^b\}$, allow us to compute the minimum mean squared error (MMSE) estimators $E(\mathbf{x}_{1..T}^a|\mathbf{y}_{1..T})$ or the maximum *a posteriori* (MAP) estimate $E(\mathbf{x}_{1..T}^a|\mathbf{y}_{1..T}, \hat{s^a}_{1..T}, \hat{s^b}_{1..T})$, where $\hat{s^a}_{1..T}, \hat{s^b}_{1..T} = \arg\max_{s_{1..T}^a, s_{1..T}^b} p(s_{1..T}^a, s_{1..T}^b|\mathbf{y}_{1..T})$, and the subscript, $1..T$, refers to all frames in the signal.

We explore two popular interaction models for which the integrals in (2) and (3) can be readily computed: *Algonquin*, and the *max model*. For signals added in the time domain, the Fourier transform of their sum is the sum of their individual Fourier transforms: $Y = X^a + X^b$. More generally, $Y = \sum_{k\in\mathcal{K}} X^k$ for a set of $N = |\mathcal{K}|$ signals. In the power spectrum,

$$|Y|^2 = \sum_{k\in\mathcal{K}} |X^k|^2 + \sum_{j\neq k} |X^j||X^k|\cos(\theta_j - \theta_k), \tag{4}$$

where θ_k is the phase of source X^k. Assuming that the phase differences are uniformly distributed:

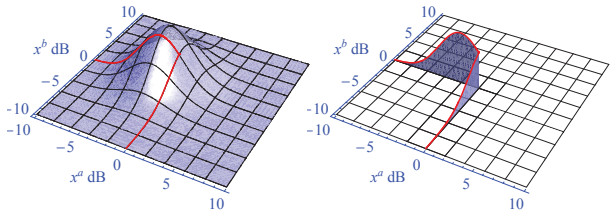
$$E\big(|Y|^2\big|\{X^k\}\big) = \sum_k |X^k|^2. \tag{5}$$

Moving the approximation into the log domain, where $x^k \triangleq \log|X^k|^2$ (and similarly for y) we have

$$y = \log\left(\sum_{k\in\mathcal{K}} \exp(x^k) + \sum_{j\neq k} \exp(\frac{x^j + x^k}{2})\cos(\theta_j - \theta_k)\right).$$

Algonquin: In the two-speaker case, Algonquin approximates this by neglecting the phase term, and using a Gaussian to model the resulting uncertainty [6]. Applying the same model to N speakers:

$$p(y|\{x^k\}) = \mathcal{N}(y; f(\{x^k\}), \psi^2), \tag{6}$$



(a) Prior density (b) Posterior density

Fig. 3. Max model: a) the prior normal density, $p(x^a|s^a)\times p(x^b|s^b)$, is shown for a single feature dimension. Its intersection with the likelihood delta function $\delta_{y-\max(x^a,x^b)}$, for $y = 0$, is represented by the red contour. b) the likelihood, $p(y = 0|s^a, s^b)$, is the integral along this contour, and the posterior, $p(x^a, x^b|y = 0, s^a, s^b)$, is the prior evaluated on this contour, normalized to integrate to one.

$$f(\{x^k\}) = \log(\sum_k \exp(x^k)). \tag{7}$$

A Newton-Laplace algorithm is used to iteratively linearize $f(\{x^k\})$, approximate both $p(y|\{s^k\})$ and the conditional posterior $p(\{x^k\}|y, \{s^k\})$ as Gaussian, and estimate the conditional expectation $E(x^k|y, \{s^k\})$. With multiple speakers the complexity of the Newton-Laplace method is $O(N^3 D_s^N)$ in each frequency band, for diagonal covariance acoustic models. Thus scaling Algonquin to larger models with more speakers is challenging.

Max model: The *max model* is an alternative to Algonquin that only requires $O(ND_s)$ computations of the univariate gaussian pdf and cumulative density functions per frequency band, followed by $O(ND_s^N)$ operations to compute $p(y|\{s_k\})$ and $E[x^k|y, \{s_k\}]$. Joint inference under the max model therefore requires $O(N^2)$ fewer operations than Algonquin.

The max model was first used in [7] for noise adaptation, where it was argued that for two additive signals, $y \approx \max(x^a, x^b)$. In [8], such a model was used to compute state likelihoods and find the optimal state sequence. Recently [9] showed that in fact $E_\theta(\mathsf{y}|x^a, x^b) = \max(x^a, x^b)$ for uniformly distributed phase. The result holds for more than two signals when $|\sum_{j\neq k} X^j| \leq |X^k|$ for any k. In general the max no longer gives the expected value, but can still be used as an approximate likelihood function:

$$p(y|\{x^k\}) = \delta_{y-\max_k\{x^k\}}, \tag{8}$$

where $\delta_{(.)}$ is a Dirac delta function.

To compute MMSE estimates of the source features using the max model requires computing $p(\{s^k\}|y)$. The max model likelihood function is piece-wise linear and so $p(\{x^k\}|y, \{s^k\})$, $p(y|\{s^k\})$, and $E(x^k|y, \{s^k\})$ all have closed-form expressions.

We follow the derivation of the posterior for two signals given in [7], and depicted in Figure 3. Define $p_{\mathsf{x}^k}(y|s^k) \triangleq p(\mathsf{x}^k = y|s^k) = \mathcal{N}(\mathsf{x}^k = y|\mu_{s^k}, \sigma_{s^k}^2)$ for random variable x^k, and the normal cumulative distribution function $\Phi_{\mathsf{x}^k}(y|s^k) \triangleq p(\mathsf{x}^k \leq y|s^k) = \int_{-\infty}^{y} \mathcal{N}(x^k; \mu_{s^k}, \sigma_{s^k}^2)\mathrm{d}x^k$. The truncated expected value is given by:

$$E(\mathsf{x}^k|\mathsf{x}^k < y, \{s^k\}) = \mu_{s^k} - \frac{\sigma_{s^k}^2 p_{\mathsf{x}^k}(y|s^k)}{\Phi_{\mathsf{x}^k}(y|s^k)}. \tag{9}$$

Since the signals are independent, the cdf of y decomposes:

$$\begin{aligned} p(\mathsf{y} \leq y|\{s^k\}) &= p(\max\{\mathsf{x}^k\} \leq y|\{s^k\}), \\ &= \prod_k \Phi_{\mathsf{x}^k}(y|s^k). \end{aligned} \tag{10}$$

The state likelihoods are then obtained by differentiating:

$$p(y|\{s^k\}) = \sum_k p_{\mathsf{x}^k}(y|s^k) \prod_{j\neq k} \Phi_{\mathsf{x}^j}(y|s^j). \tag{11}$$

From this we readily see that the individual terms in the above sum correspond to $p(\mathsf{y} = y, \mathsf{x}^k = y|\{s^k\})$. The conditional probability that source k is maximum then is:

$$\pi_k \triangleq p(\mathsf{x}^k = y|\mathsf{y} = y, \{s^k\}) = \left(\sum_j \frac{p_{\mathsf{x}^j}(y|s^j)}{\Phi_{\mathsf{x}^j}(y|s^j)}\right)^{-1} \frac{p_{\mathsf{x}^k}(y|s^k)}{\Phi_{\mathsf{x}^k}(y|s^k)}.$$

The expected value of each signal given the observation and states can now be written using (9)

$$\begin{aligned} E(\mathsf{x}^k|y, \{s^k\}) &= \pi_k y + (1-\pi_k) E(\mathsf{x}^k|\mathsf{x}^k < y, \{s^k\}), \\ &= \pi_k y + (1-\pi_k)\left(\mu_k - \frac{\sigma_k^2 p_{\mathsf{x}^k}(y|s^k)}{\Phi_{\mathsf{x}^k}(y|s^k)}\right). \end{aligned}$$

The loopy belief propagation algorithm presented in this paper requires that the marginal likelihoods $p(\mathbf{y}|s^k) = \sum_{s^j\neq s^k} \prod_{j\neq k} p(s^j) \prod_f p(y_f|\{s^i\})$ be iteratively computed for each source. In general this computation requires at least $O(D_s^N)$ operations per source, because all possible combinations of acoustic states must be considered. This is the case for both Algonquin and the max model. Under the max model, however, the data likelihood in a single frequency band (11) consists of N terms, each of which *factor* over the acoustic states of the sources. Currently we are investigating linear-time algorithms ($O(ND_s)$) that exploit this property to approximate $p(\mathbf{y}|s^k)$.

In many combinations of states one model may be significantly louder than the others $\mu_{s^k} \gg \mu_{\{s^{j\neq k}\}}$ in a given frequency band, relative to their variances. In such cases we can closely approximate the likelihood as $p(y|\{s^k\}) \approx p_{\mathsf{x}^k}(y|s^k)$, and the posterior expected values according to $E(\mathsf{x}^k = y|y, \{s^k\}) \approx y$ and $E(\mathsf{x}^k < y|y, \{s^k\}) \approx \min(y, \mu_{s^k})$. This results in a significantly faster algorithm. In our experiments the approximation made no significant difference in accuracy and is therefore used in place of the exact max algorithm.

4. INFERENCE

In [2] exact inference was done in this model using a 2–D Viterbi search on the product model HMM shown in figure 2(b). Given the most likely state sequences of both speakers, MMSE estimates of the sources can be computed using Algonquin or the max model, and averaging over acoustic states. Once the log spectrum of each source is estimated, the corresponding time-domain signal can be recovered using the phase of the mixture features.

The exact inference algorithm is derived by combining the state variables into the joint states $s_t = (s_t^a, s_t^b)$ and $v_t = (v_t^a, v_t^b)$. The model can then be treated as a single hidden Markov model with transitions given by $p(v_t^a|v_{t-1}^a) \times p(v_t^b|v_{t-1}^b)$, and likelihoods from Eqn. (1). However inference in such a factorial HMM is more efficient if a two–dimensional Viterbi search is used to find the most likely joint state sequences $v_{1..T}^a$, $v_{1..T}^b$. With N speakers, the corresponding N–D Viterbi algorithm has complexity $O(ND_v^{N+1})$ per frame, where D_v is the number of grammar states [2]. In practice the complexity is somewhat less than this due to the sparseness of the grammar and the use of state pruning, or *beam search*.

Belief Propagation: To avoid the combinatorial explosion of exact inference, which scales exponentially with the number of speakers N, we can iteratively estimate the configurations of the speakers. Using the max-product belief propagation method [4, 10], temporal inference can be accomplished with complexity $O(TND_v^2)$.

The max-product algorithm can be viewed as a generalization of the Viterbi algorithm to arbitrary graphs of random variables. For any probability model defined on a set of random variables $x \triangleq \{x_i\}$:

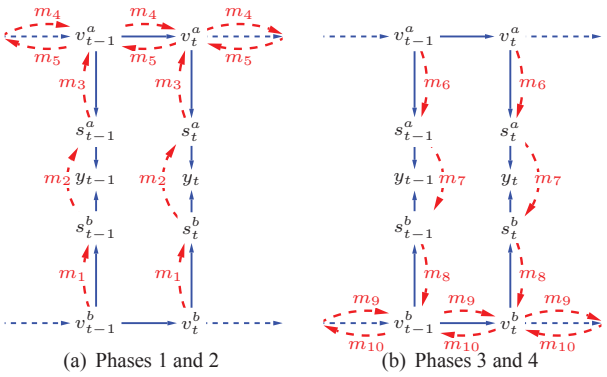
$$p(x) \propto \prod_{\mathcal{C}\in\mathcal{S}} f_{\mathcal{C}}(x_{\mathcal{C}}), \tag{12}$$

where the factors $f_{\mathcal{C}}(x_{\mathcal{C}})$ are defined on subsets of variables $x_{\mathcal{C}} \triangleq \{x_i : i \in \mathcal{C}\}$, and $\mathcal{S} = \{\mathcal{C}\}$. Inference using the algorithm consists of iteratively passing messages between "connected" random variables of the model. For a given random variable x_i, the message from variable set $x_{\mathcal{C}\setminus i} \triangleq \{x_j : i \in \mathcal{C}, j \neq i \in \mathcal{C}\}$ to x_i is:

$$m_{x_{\mathcal{C}\setminus i}\to x_i}(x_i) = \max_{x_{\mathcal{C}\setminus i}} f_{\mathcal{C}}(x_{\mathcal{C}}) \prod_{j\in\mathcal{C}\setminus i} \frac{q(x_j)}{m_{x_{\mathcal{C}\setminus j}\to x_j}(x_j)}, \tag{13}$$

$$\hat{x}_{\mathcal{C}\setminus i}(x_i) = \arg\max_{x_{\mathcal{C}\setminus i}} f_{\mathcal{C}}(x_{\mathcal{C}}) \prod_{j\in\mathcal{C}\setminus i} \frac{q(x_j)}{m_{x_{\mathcal{C}\setminus j}\to x_j}(x_j)}, \tag{14}$$

where $\hat{x}_{\mathcal{C}\setminus i}(x_i)$ stores the maximizing configuration of $x_{\mathcal{C}\setminus i}$ for each x_i, and $q(x_i) = \prod_{\mathcal{C}:i\in\mathcal{C}} m_{x_{\mathcal{C}\setminus i}\to x_i}(x_i)$ is the product of all messages to variable x_i from neighboring variables.



(a) Phases 1 and 2 (b) Phases 3 and 4

Fig. 4. Message passing sequences ($m_1 \ldots m_{10}$). The messages shown in a chain, such as m_4, are passed sequentially along the entire chain, in the direction of the arrows, before moving to the next message. Note that messages m_6 through m_{10} are the same as m_1 through m_5, but with a and b swapped.

Optimization consists of passing messages according to a *message passing schedule*. When the probability model is tree-structured, the global MAP configuration of the variables can be found by propagating messages up and down the tree, and then "decoding", by recursively evaluating $\hat{x}_{\mathcal{C}\setminus i}(x_i)\ \forall\ \mathcal{C} : i \in \mathcal{C}$, starting from any x_i. When the model contains loops, as do the models we consider here, the messages must be iteratively updated because there are cycles in the graph, and there is no guarantee that this approach will converge to the MAP configuration. However, if the algorithm converges, the MAP estimate is guaranteed to be a local MAP configuration over a potentially exponentially large neighborhood [10].

A natural message-passing schedule is to alternate between passing messages from one grammar chain to the other, and along the grammar chain of the receiving source, as shown in Figure 4. All messages are initialized to be uniform, and v_1^a and v_1^b are initialized to their priors.

There are four phases of inference:
1. Pass messages from source b to source a through the interaction function $p(\mathbf{y}_t|s_t^a, s_t^b)$ for all t (messages m_1-m_3):

$$m_1(s_t^b) \triangleq m_{v_t^b \to s_t^b} = \max_{v_t^b} p(s_t^b|v_t^b) m_{v_{t-1}^b \to v_t^b} m_{v_{t+1}^b \to v_t^b}$$

$$m_2(s_t^a) \triangleq m_{s_t^b \to s_t^a} = \max_{s_t^b} p(\mathbf{y}_t|s_t^a, s_t^b) m_{v_t^b \to s_t^b}$$

$$m_3(v_t^a) \triangleq m_{s_t^a \to v_t^a} = \max_{s_t^a} p(s_t^a|v_t^a) m_{s_t^b \to s_t^a}$$

2. Pass messages forward along the grammar chain for source a, for $t = 1..T$, and then backward, for $t = T..1$ (messages m_4-m_5):

$$m_4(v_t^a) \triangleq m_{v_{t-1}^a \to v_t^a} = \max_{v_{t-1}^a} p(v_t^a|v_{t-1}^a) m_{v_{t-2}^a \to v_{t-1}^a} m_{s_{t-1}^a \to v_{t-1}^a}$$

$$m_5(v_t^a) \triangleq m_{v_{t+1}^a \to v_t^a} = \max_{v_{t+1}^a} p(v_{t+1}^a|v_t^a) m_{v_{t+2}^a \to v_{t+1}^a} m_{s_{t+1}^a \to v_{t+1}^a}$$

3. Pass messages from source b to a for all t, (messages m_6-m_8).
4. Pass messages forward along the grammar chain for source b, for $t = 1..T$, and then backward, for $t = T..1$ (messages m_9-m_{10}). Note that the max-product algorithm also decouples the interaction between the acoustic and grammar states across sources. Naively this complexity would be $O(ND_s^N D_v^N)$. Given the factorized structure of the model, the complexity reduces to $O(\sum_{k=1}^N D_s^{N-k+1} D_v^k) \leq O(ND^{N+1})$, where $D = \max(D_s, D_v)$. In the max-product algorithm, the complexity is further reduced to $O(ND_sD_v)$ per iteration (see messages 1, 3, 6, 8).

5. EXPERIMENTS

Table 1 summarizes the error rate of our multi-talker speech recognition system on the SSC task [1], as a function of separation algorithm. In all cases, oracle speaker identities and gains were used to define the speaker-dependent acoustic models used during separation. Recognition was done on the reconstructed target signal using a conventional single-talker speech recognition system that does speaker-dependent labeling [2].
For all iterative algorithms, the message passing schedule was executed for 10 iterations. After inferring the grammar state sequences, conditional MMSEs of the sources were reconstructed.
For the *max-sum product algorithm*, the max operations in the messages sent between the sources are replaced with sums. The *iterative Viterbi* algorithm is equivalent to the max-sum product algorithm, but with the grammar to acoustic messages bottlenecked to the single maximum value.
The max-sum-product algorithm produces nearly the same accuracy as exact inference. The results obtained using the max-sum product algorithm are significantly better than those of the max-product algorithm, presumably because this leads to more accurate grammar state likelihoods. The max-sum product algorithm is an order of magnitude faster than the exact temporal inference, and still exceeds the average performance of human listeners on the task. As seen in Table 2, even for two sources, temporal inference with loopy belief propagation is three times more efficient than joint-Viterbi with a beam of 400, which yields comparable task error rates. The approach is promising because temporal inference scales *linearly* with language model size, and *linearly* with the number of sources, making it applicable to more complex problems.

Conditon	Humans	Joint Viterbi	Max Product	Iterative Viterbi	Max-Sum Product
ST	34.0	**33.3**	42.0	44.3	39.7 (38.6)
SG	19.5	**11.5**	**12.9**	**16.4**	**12.0 (14.4)**
DG	11.9	**9.9**	12.0	13.9	**11.1 (10.8)**
Overall	22.3	**19.0**	23.3	25.8	**21.9 (22.1)**

Table 1. SSC task error rate as a function of separation algorithm and test condition. Conditions are: *same talker* (ST), *same gender* (SG), *different gender* (DG). In all cases Algonquin was used to approximate the acoustic likelihoods. Max interaction results are in parentheses. Results exceeding human performance are bolded.

Algorithm	Joint Viterbi	Joint Viterbi	Max-Sum Product	
Likelihoods	Algonquin	Algonquin	Algonquin	Max
Beam size	20000	400	Full	Full
Error Rate	19.0	22.1	21.9	22.1
Relative Operations	$10n$	$3n$	n	n

Table 2. Task error rate and relative number of operations required for temporal inference as a function of algorithm, likelihood model, and beam size.

6. REFERENCES

[1] M. Cooke, J. Hershey, and S. Rennie, "The speech separation and recognition challenge," *Computer Speech and Language (to appear)*, 2009.

[2] J. Hershey, T. Kristjansson, S. Rennie, and P. Olsen, "Single channel speech separation using layered hidden Markov models," *NIPS*, pp. 593–600, 2006.

[3] Y. Weiss, "Interpreting images by propagating bayesian beliefs," *NIPS*, pp. 908–915, 1997.

[4] F. Kschischang, B. Frey, and H. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. on Info. Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[5] M. Reyes-Gómez, N. Jojic, and D. Ellis, "Towards single-channel unsupervised source separation of speech mixtures: The layered harmonics/formants separation/tracking model," in *Workshop on Statistical and Perceptual Audio Processing*, 2004.

[6] B. Frey, T. Kristjansson, L. Deng, and A. Acero, "Algonquin - learning dynamic noise models from noisy speech for robust speech recognition," *NIPS*, pp. 1165–1171, 2001.

[7] A. Nádas, D. Nahamoo, and M. Picheny, "Speech recognition using noise-adaptive prototypes," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 1495–1503, 1989.

[8] P. Varga and R.K. Moore, "Hidden Markov model decomposition of speech and noise," *ICASSP*, pp. 845–848, 1990.

[9] M.H. Radfar, R.M. Dansereau, and A. Sayadiyan, "Nonlinear minimum mean square error estimator for mixture-maximisation approximation," *Electronics Letters*, vol. 42, no. 12, pp. 724–725, 2006.

[10] Y. Weiss and W. Freeman, "On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs," *IEEE Trans.on Info. Theory*, vol. 47, no. 2, pp. 736–744, 2001.