Sum-Product Networks for Structured Prediction: Context-Specific Deep Conditional Random Fields

Martin Ratajczak MARTIN.RATAJCZAK@TUGRAZ.AT

Sebastian Tschiatschek TSCHIATSCHEK@TUGRAZ.AT

Franz Pernkopf PERNKOPF@TUGRAZ.AT
Graz University of Technology, 8010 Graz, Inffeldgasse 16c

Abstract

Linear-chain conditional random fields (LC-CRFs) have been successfully applied in many structured prediction tasks. Many previous extensions, e.g. replacing local factors by neural networks, are computationally demanding. In this paper, we extend conventional LC-CRFs by replacing the local factors with sum-product networks, i.e. a promising new deep architecture allowing for exact and efficient inference. The proposed local factors can be interpreted as an extension of Gaussian mixture models (GMMs). Thus, we provide a powerful alternative to LC-CRFs extended by GMMs. In extensive experiments, we achieved performance competitive to state-of-the-art methods in phone classification and optical character recognition tasks.

1. Introduction

Structured prediction and in particular sequence labeling are core components in many applications, e.g. speech recognition (Gunawardana et al., 2005) and natural language processing (Fosler-Lussier et al., 2013). Many of these applications have been successfully solved by discriminative and probabilistic approaches such as *maximum entropy Markov models (MEMMs)* (McCallum et al., 2000) and *linear-chain conditional random fields (LC-CRFs)* (Lafferty et al., 2001). Both approaches have several advantages yielding better performance than their generative counterparts, i.e. hidden Markov models (HMMs): First, observed variables can be dependent given the label sequence. This increases the model's expressiveness compared to HMMs which assume conditional independence between the observations given the label sequence. Second, the discriminative objective function, the conditional likelihood, directly optimizes the relationship between input and output variables. That is, the conditional likelihood focuses on the prediction of the best output (one label or label sequence) instead of estimating the joint probability distribution over the output and input variables. Third, the negative conditional likelihood, is convex in the model weights for arbitrary but fixed feature functions. Fourth, in case of LC-CRFs, normalization is performed over the whole output sequence and not locally in contrast to HMMs and MEMMs. This counteracts the *label bias* problem. Nevertheless, MEMMs are of interest in various applications as they can be easily extended to arbitrary long histories and have lower time complexity in training.

MEMMs and LC-CRFs consist of transition factors, modeling the relationship between the output labels, and local factors, modeling the relationship between input observations and output labels. Several approaches have been proposed to parametrize and to learn the non-linear feature functions of the local factors. One popular choice is replacing the local factors by multi-layer neural networks (Peng et al., 2009; Prabhavalkar & Fosler-Lussier, 2010). In contrast to this approach, there are models which represent a probability distribution over the output *and* the hidden variables and allow for exact and efficient inference. A prominent example is the Gaussian mixture model (GMM) which has been applied extensively for many years in conjunction with HMMs and LC-CRFs (Fosler-Lussier et al., 2013) because of its scalability. Another approach is the *hidden-unit conditional random field* (HU-CRF) (van der Maaten et al., 2011) which extends the LC-CRF by replacing the local factors with the *discriminative RBM* (DRBM) (Larochelle & Bengio, 2008). Unfortunately, the HU-CRF is limited to a single hidden layer but exact inference is efficient.

Most probabilistic deep models require approximate algorithms for efficient training, e.g. contrastive divergence (Hinton, 2002). However, there is evidence in-

Proceedings of the 31^{st} *International Conference on Machine Learning*, Beijing, China, 2014. JMLR: W&CP volume 32. Copyright 2014 by the author(s).

dicating that approximate inference can make it more difficult to learn structured models and even lead to inferior results (Kulesza & Pereira, 2007). In contrast to the former approaches, *sum-product networks* (SPNs) (Poon & Domingos, 2011) enable efficient and exact training of deep models with many hidden layers. The *discriminative SPN* outperformed deep neural networks and other methods on a difficult image classification task (Gens & Domingos, 2012).

In this paper, we represent the local factors of LC-CRFs by a specific type of sum-product networks, enabling exact *and* efficient inference of potentially deep models. These local factors are called *context-specific deep CRFs (CS-DCRFs)*, i.e. conditional undirected graphical models with given input variables, multiple layers of hidden variables and one output variable. We emphasize the model's relation to *context-specific undirected graphical models* with *higher order factors* (Tarlow et al., 2010; Nyman et al., 2013). While context-specific undirected graphical models have not received much attention, their directed counterpart, i.e. Bayesian networks, have been introduced many years ago (Boutilier et al., 1996; Friedman et al., 1997). In contrast to typical deep RBMs (Hinton, 2002), CS-DCRFs go beyond pairwise factors and model the relationship between variables in multiple layers from the top to the lowest layer. In general, exact inference in such models is intractable. Only by restricting the model structure and the local factors to context-specific factors enables exact and efficient inference.

The main contributions of our work are:
(i) *Extension of LC-CRFs and MEMMs* by deep local factors, i.e. CS-DCRFs. These models are applied to *structured prediction*, in particular, *sequence labeling*. Experimental results for phone classification and handwriting recognition are presented.
(ii) Usage of the *forward-backward algorithm* for both the *deep architecture* and the *LC-CRF*. As a consequence, *exact inference* is efficient and *joint training* of the deep model and the LC-CRF using the *discriminative training criterion* is enabled.

The remainder of this paper is structured as follows: In Section 2 we briefly review related work. In Section 3 we introduce the CS-DCRF models and discuss their representation as context-specific undirected graphical models with higher order factors and as sum-product networks. We present the classifier model first and then we extend MEMMs and LC-CRFs. In Section 4 we evaluate these models on sequence labeling tasks in optical character recognition and phone classification. Finally, we conclude our paper and point out future work in Section 5.

2. Other Related Work

Discriminative SPNs have been introduced in Gens & Domingos (2012). Our work differs in several points. First, we formulate our model in a different way which is not based on Darwiche's network polynomial (Darwiche, 2000; 2003). Second, we utilize message passing to compute the model's marginal probabilities in contrast to back-propagation (Poon & Domingos, 2011; Gens & Domingos, 2012). This alternative formulation as message passing is in particular interesting, since the forward-backward algorithm is very popular and well known from HMMs and its variants. Third, to the best of our knowledge, in Gens & Domingos (2012) the model weights in the lowest layer are fixed. In contrast, we train all model weights. Fourth, we summed out the hidden variables in all our experiments in contrast to using the maximum approximation (Poon & Domingos, 2011; Gens & Domingos, 2012). Last but not least, we targeted structured prediction in contrast to single label classification task.

In previous work, deep architectures have been used in LC-CRFs. One approach is to pre-train a deep belief network on the input data in an unsupervised way. The deep belief network is then transformed into a multi-layer neural network with sigmoid activations and plugged into the LC-CRF (Do & Artières, 2010). Keeping the deep model fixed, the LC-CRF is pre-trained in a supervised way. Finally, the whole model, i.e. the LC-CRF and the deep model, is fine-tuned by back-propagation. Other approaches, such as conditional neural fields (CNFs) (Peng et al., 2009) and multi-layer CRFs (Prabhavalkar & Fosler-Lussier, 2010), propose a direct method to optimize multi-layer neural networks and LC-CRFs by the conditional likelihood criterion based on error back-propagation. In these approaches, the local factors in LC-CRFs are extended by one or more layers of hidden neurons with deterministic non-linear activation functions.

A special case of our CS-DCRF can be interpreted as discriminative GMM with class-independent and component-independent covariance matrix, if the model is restricted to one hidden layer and only one hidden component variable. The covariance matrix drops out.

3. Context-Specific Deep CRF

First, in Section 3.1 we present a CS-DCRF classifier represented as context-specific conditional undirected graphical model and as sum-product network. Second, in Section 3.2 and 3.3 we integrate this model into MEMMs and LC-CRFs, respectively.

3.1. CS-DCRF Classifier

Model Definition. The probability distribution of an undirected graphical model is defined as the product over a

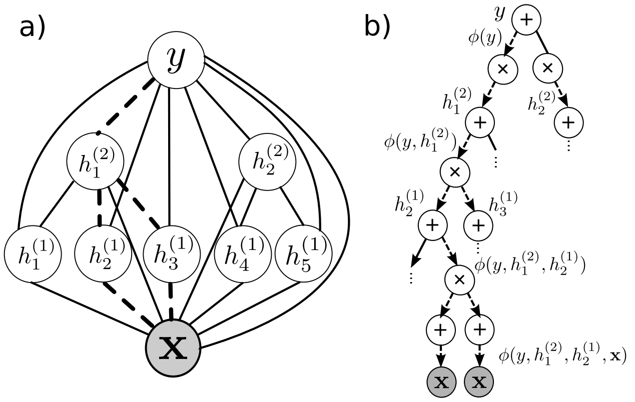

Figure 1. Context-specific deep CRF represented as a) conditional undirected graphical model and as b) sum-product network. Dashed edges indicate the involved variables of the higher order factors.

set of clique factors $\phi_k(\cdot)$ and subsequent normalization. In this way, we define our classifier model by the probability distribution

$$p(y, \mathbf{h}|\mathbf{x}) = \frac{\prod_k \phi_k(y, \mathbf{h}, \mathbf{x})}{Z(\mathbf{x})} \tag{1}$$

over the output variable y (class label) *and* a set of hidden variables $\mathbf{h}$ given a set of input variables $\mathbf{x}$. The set of hidden variables $\mathbf{h} = \{\mathbf{h}^{(1)}, \ldots, \mathbf{h}^{(L)}\}$ is the union of the hidden variables $\mathbf{h}^{(l)}$ over L hidden layers and $Z(\mathbf{x})$ is the partition function, i.e. normalization constant. Marginalizing the model posterior $p(y, \mathbf{h}|\mathbf{x})$ over the hidden variables $\mathbf{h}$ determines the probability distribution

$$p(y|\mathbf{x}) = \frac{Q(y, \mathbf{x})}{Z(\mathbf{x})}, \tag{2}$$

where $Q(y, \mathbf{x}) = \sum_{\mathbf{h}} \prod_k \phi_k(y, \mathbf{h}, \mathbf{x})$ and $Z(\mathbf{x}) = \sum_y Q(y, \mathbf{x})$. Without further assumptions, computing the partition function is intractable. Therefore, we restrict our model to a specific model structure to enable efficient inference.

Model Structure. A special instance of that model with two hidden layers is shown in Figure 1a, represented as an undirected graphical model. The nodes in the graph represent input variables, multiple layers of hidden variables and one output variable. The edges represent direct dependencies between variables. The restrictions to our model are: First, no edges connect the hidden variables within the same layer similar to RBMs (Hinton, 2002). Second, hidden variables must not have edges to more than one hidden variable in the layer above (its parent). Third, hidden variables connect not only to its parent but also to the parent of their parent and so on. This way, our model represents *higher order factors* going beyond pairwise factors usually used in RBMs. In Figure 1a, the set of variables $\{y, h_1^{(2)}, h_2^{(1)}, \mathbf{x}\}$ forms cliques in the graphical model and are modeled by corresponding bias, pairwise and *higher order factor functions*: $\phi(y)$, $\phi(y, h_1^{(2)})$, $\phi(y, h_1^{(2)}, h_2^{(1)})$ and $\phi(y, h_1^{(2)}, h_2^{(1)}, \mathbf{x})$.

Context-specific Factors. The value of the higher order factor $\phi(k^{(l)})$ is determined by a set of pairs

$$k^{(l)} := \bigcup_{l'=L+1:l} \{(i^{(l)}, h_i^{(l)})\},$$

where the index variables $i^{(l)}$ select one hidden variable per layer (or the class label in the top layer) and $h_i^{(l)}$ is its value. Formally, the context $c^{(l)}$ of the variables $i_c^{(l)}$ and $h_{i,c}^{(l)}$ in layer l is the set of all index variables in the same clique and its values excluding the variables itself and its values

$$c^{(l)} := k^{(l)} \setminus \{(i_c^{(l)}, h_{i,c}^{(l)})\} = k^{(l+1)}, \tag{3}$$

where the top layer $L+1$ has empty context $c^{(L+1)} := \{\varnothing\}$ and the layer L has the output variable as context $c^{(L)} := \{y\}$. The context-specific index variable $i_c^{(l)}$ denotes the index variable $i^{(l)} \in \mathcal{I}_c^{(l)}$. $\mathcal{I}_c^{(l)} \subset \mathcal{I}^{(l)}$ is restricted to a context-specific index set where $\mathcal{I}^{(l)}$ is an index set enumerating all the hidden variables in layer l. Similarly, the context-specific hidden variable $h_{i,c}^{(l)}$ denotes the hidden variable $h_i^{(l)} \in \mathcal{H}_{i,c}^{(l)}$. $\mathcal{H}_{i,c}^{(l)} \subset \mathcal{H}_i^{(l)}$ is restricted to the context-specific state space where $\mathcal{H}_i^{(l)}$ is the state space of the hidden variable $h_i^{(l)}$. In the sense of the above context definition, we further restricted our model to context-specific factors, i.e. the value of the higher order factor $\phi(k^{(l)})$ is non-constant only for particular configurations of the context $c^{(l)}$, otherwise it is set to one.

Sum-product Form. The summation over the hidden variables in the function $Q(y, \mathbf{x})$ can be reordered so that each factor function belongs to one corresponding summation which can be written as product of summations. Consequently, the function $Q(y, \mathbf{x})$ is specified as

$$\begin{aligned} Q(y, \mathbf{x}) = &\phi(y) \prod_{i_c^{(L)}} \sum_{h_{i,c}^{(L)}} \phi(c^{(L)}, h_{i,c}^{(L)}) \ldots \\ &\prod_{i_c^{(1)}} \sum_{h_{i,c}^{(1)}} \phi(c^{(l)}, h_{i,c}^{(1)}) \prod_{i_c^{(0)}} \phi(c^{(0)}, i_c^{(0)}, \mathbf{x}) \end{aligned} \tag{4}$$

i.e. products and weighted summations are alternated avoiding an exhaustive summation over the whole state space. We omitted to sum over factor functions with constant value of one. We abbreviated the higher order factors $\phi(y, h_1^{(2)}, h_2^{(1)})$ and $\phi(y, h_1^{(2)}, h_2^{(1)}, \mathbf{x})$ following the definition of context by $\phi(c^{(l)}, h_{i,c}^{(1)})$ and $\phi(c^{(0)}, i_c^{(0)}, \mathbf{x})$, respectively, in the former example in Figure 1a. The computation of the function $Q(y, \mathbf{x})$ and the partition function $Z(\mathbf{x})$ can be represented by a *sum-product network* (Poon & Domingos, 2011) as illustrated in Figure 1b. Weighted summations are represented as sum nodes, products as product nodes and the input variables as filled leave nodes. The edges of the sum nodes hold the weights representing the higher order factors. The dashed edges represent a particular subset of factors also shown in the graphical model in Figure 1a.

Model Parametrization. We parametrize our model as a log-linear model which is optimal with regard to the maximum entropy criterion under moment constraints (Berger et al., 1996) so the probability distribution of the model posterior is specified by the Gibbs distribution

$$p(y, \mathbf{h}|\mathbf{x}) = \frac{\exp(\sum_k w_k f_k(y, \mathbf{h}, \mathbf{x}))}{Z(\mathbf{x})}.$$

The higher order factors $\phi_k(\cdot) = \exp(w_k\, f_k(\cdot))$ have corresponding weights w_k and feature functions $f_k(\cdot)$. The feature functions for the lowest and the remaining layers are $f_m(\cdot) = \delta(m, m'(y, \mathbf{h}))\hat{f}_m(\mathbf{x})$, where $m = c^{(l)} \cup \{i_c^{(l)}\}$ and $\hat{f}_m(\mathbf{x})$ is an arbitrary feature function, and $f_k(\cdot) = \delta(k, k'(y, \mathbf{h}))$, respectively, where $k = c^{(l)} \cup \{(i_c^{(l)}, h_{i,c}^{(l)})\}$.

Model Optimization. The model weights $\mathbf{w} = (w_k)$ are optimized to maximize the logarithm of the conditional likelihood over the training set, i.e.

$$F(\mathbf{w}, \mathcal{D}) = \sum_{n=1}^{N} \log p(y_n|\mathbf{x}_n),$$

where $\mathcal{D} = \{(y_1, \mathbf{x}_1), \ldots, (y_N, \mathbf{x}_N)\}$ is a given labeled training set drawn i.i.d. from an unknown data distribution. To optimize the objective by first-order gradient ascent methods, we need to compute the partial derivatives of $F(\mathbf{w}, \mathcal{D})$ with respect to the weights. The gradients of the top layer are

$$\frac{\partial F}{\partial w_y} = \sum_{n=1}^{N} \delta(y_n, y) - p(y|\mathbf{x}_n).$$

Furthermore, the gradients of each hidden layer l are

$$\frac{\partial F}{\partial w_{k^{(l)}}} = \sum_{n=1}^{N} \delta(y_n, y)p(k^{(l)} \setminus y|y_n, \mathbf{x}_n) - p(k^{(l)}|\mathbf{x}_n), \tag{5}$$

where $\delta(y_n, y)$ is the indicator function. These gradients represent the difference between the empirical and model expectation of the corresponding feature functions as in log-linear models (Berger et al., 1996). The marginal probabilities for the higher order factors

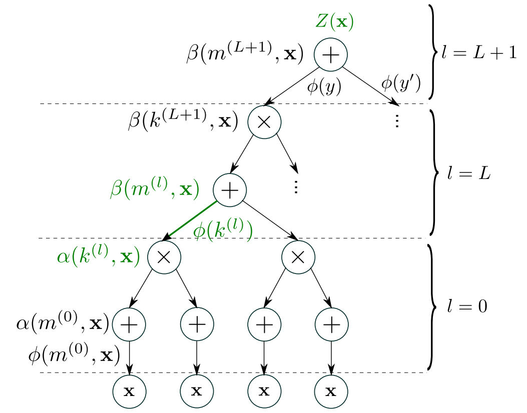
$Z(\mathbf{x})$
$\beta(m^{(L+1)}, \mathbf{x})$
$\phi(y)$
$\phi(y')$
$l = L + 1$
$\beta(k^{(L+1)}, \mathbf{x})$
$l = L$
$\beta(m^{(l)}, \mathbf{x})$
$\phi(k^{(l)})$
$\alpha(k^{(l)}, \mathbf{x})$
$l = 0$
$\alpha(m^{(0)}, \mathbf{x})$
$\phi(m^{(0)}, \mathbf{x})$
x
x
x
x

Figure 2. CS-DCRF with single hidden layer $L = 1$ illustrating the forward-backward algorithm. The marginals of the higher order factors can be computed by multiplying the forward message $\alpha(.)$, the backward message $\beta(.)$ and the corresponding factor $\phi(.)$ divided by the partition function $Z(\mathbf{x})$ (indicated by green color).

$$p(k^{(l)}|\mathbf{x}) = \frac{\beta(m^{(l)}, \mathbf{x})\phi(k^{(l)})\alpha(k^{(l)}, \mathbf{x})}{Z(\mathbf{x})} \tag{6}$$

$$p(k^{(l)} \setminus y|y, \mathbf{x}) = \frac{\beta(m^{(l)}, \mathbf{x})\phi(k^{(l)})\alpha(k^{(l)}, \mathbf{x})}{Q(y, \mathbf{x})} \tag{7}$$

are efficiently computed by the forward-backward algorithm. Details are presented in the next paragraph. The gradients of the lowest layer ($l = 0$) are

$$\frac{\partial F}{\partial w_{m^{(0)}}} = \sum_{n=1}^{N} f_{m^{(0)}}(\mathbf{x}_n) \Big[\delta(y_n, y)p(k^{(0)} \setminus y|y_n, \mathbf{x}_n) - p(k^{(0)}|\mathbf{x}_n)\Big]. \tag{8}$$

Forward-backward Algorithm. Figure 2 illustrates the *forward-backward algorithm* which enables the efficient computation of the partition functions $Q(y, \mathbf{x})$ and $Z(\mathbf{x})$ as well as the marginal probabilities. Passing the messages from the leaves to the root node computes the partition function.

Table 1 summarizes the forward-backward algorithm. The forward messages are initialized with the observation-dependent factors and then we calculate alternating products and weighted summations for all layer l from bottom to the top layer according to the recursions in Table 1. Finally, in the top layer $L + 1$ the value of the partition function $Z(\mathbf{x}) = \alpha(i^{(L+1)}, \mathbf{x})$ is stored. The backward recursion proceeds in a similar way. The backward recursion is initialized with the value of 1 in the top layer. Then for

Table 1. Forward-backward algorithm.

initialize forward step	
$\alpha(m^{(0)}, \mathbf{x})$	$= \phi(m^{(0)}, \mathbf{x})$
$\alpha(k^{(1)}, \mathbf{x})$	$= \prod_{i_c^{(0)}} \alpha(m^{(0)}, \mathbf{x})$
forward pass: for each layer $l = 1, ..., L+1$	
$\alpha(m^{(l)}, \mathbf{x})$	$= \sum_{h_{i,c}^{(l)}} \phi(k^{(l)}) \alpha(k^{(l)}, \mathbf{x})$
$\alpha(k^{(l+1)}, \mathbf{x})$	$= \prod_{i_c^{(l)}} \alpha(m^{(l)}, \mathbf{x})$
initialize backward step	
$\beta(m^{(L+1)}, \mathbf{x})$	$= 1$
backward pass: for each layer $l = L, ..., 1$	
$\beta(k^{(l)}, \mathbf{x})$	$= \phi(k^{(l+1)}) \beta(m^{(l+1)}, \mathbf{x})$
$\beta(m^{(l)}, \mathbf{x})$	$= \beta(k^{(l)}, \mathbf{x}) \prod_{\tilde{i}_c^{(l)} \setminus i_c^{(l)}} \alpha(\tilde{k}^{(l)}, \mathbf{x})$
where $m^{(l)} = c^{(l)} \cup \{i_c^{(l)}\}$ and $k^{(l)} = c^{(l)} \cup \{(i_c^{(l)}, h_{i,c}^{(l)})\}$	

each state of the hidden variable $h_{i,c}^{(l)}$ the backward messages $\beta(m^{(l)}, \mathbf{x})$ from the layer above are weighted by the factor $\phi(k^{(l)})$. In the next step we calculate the product of the forward messages $\alpha(\tilde{k}^{(l)})$ over the indices $\tilde{i}_c^{(l)}$ omitting the index $i_c^{(l)}$ and weight this product by the backward message of the above layer.

Time Complexity. We obtain the time complexity $\mathcal{O}(YI^{(0)}(IH)^L)$ of the forward-backward algorithm assuming equal cardinality in each layer l for the context-specific state space of hidden variables H and index set I, where Y is the number of class labels and $I^{(0)}$ is the number of feature functions in the lowest layer. The time complexity grows only exponentially with the number of hidden layers L and polynomial in I and H. In contrast, the time complexity of RBMs grows in the number of possible states of the hidden variables $\mathcal{O}(LH^{2I})$ (Hinton, 2002) which is intractable.

3.2. MEMM augmented by CS-DCRF

In this section, we extend higher order MEMMs by CS-DCRFs local factors. We consider sequence labeling problems with the aim of assigning a sequence of labels given an input sequence.

Model Definition. Higher order MEMMs model the conditional probability of one label y_t at sequence index t given the $M-1$ previous labels $h_t = y_{t-M+1:t-1}$ and the observed sequence $\mathbf{x}_{1:T}$, i.e.

$$p(y_t|h_t, \mathbf{x}_{1:T}) = \frac{\phi(y_t, \mathbf{x}_{1:T})\phi(h_t, y_t)}{Z(h_t, \mathbf{x}_{1:T})}, \qquad (9)$$

where T is the sequence length. The relationship between the label history h_t and y_t is modeled by transition factors $\phi(h_t, y_t)$. Further, the relationship between the input variables and labels at sequence index t is described by local factors $\phi(y_t, \mathbf{x}_{1:T})$. MEMMs are locally normalized, i.e. the partition function is computed as

$$Z(h_t, \mathbf{x}_{1:T}) = \sum_{y_t} \phi(y_t, \mathbf{x}_{1:T})\phi(h_t, y_t). \qquad (10)$$

The conditional probability of the sequence labels $y_{1:T}$ given the observed sequence $\mathbf{x}_{1:T}$ is

$$p(y_{1:T}|\mathbf{x}_{1:T}) = \prod_{t=1}^{T} p(y_t|h_t, \mathbf{x}_{1:T}). \qquad (11)$$

Gradients. The higher order Markov assumption and the local normalization enable to efficientlly compute all marginals and the following gradients without the forward-backward algorithm, cf. Section 3.3.

We extend MEMMs by replacing the local factors in Eq. (9) by CS-DCRFs, i.e. $\phi(y_t, \mathbf{x}_{1:T}) = \alpha_t^{local}(y_t, \mathbf{x}_{1:T}) = \phi(y_t)\alpha^{deep}(y_t, \mathbf{x}_{1:T})$. The gradients for the transition features are

$$\frac{\partial F}{\partial w_{k,y_t}} = \sum_{n=1}^{N}\sum_{t=1}^{T} f_k(h_{t,n})\Big[\delta(y_{t,n}, y_t) - p(y_t|\, h_{t,n}\,, \mathbf{x}_{1:T,n})\Big],$$

where $k = y_{t-m}$ and $f_k(h_{t,n}) = \delta(y_{t-m,n}, y_{t-m})$ are distant bigram feature functions for all $m = 1, ..., M-1$ previous labels.

Inference. For details on how to compute the most probable sequence

$$\hat{y}_{1:T} = \operatorname*{argmax}_{y_{1:T}} \prod_{t=1}^{T} p(y_t|\, h_t\,, \mathbf{x}_{1:T})$$

for first order MEMMs (M=1) using the Viterbi algorithm, we refer to McCallum et al. (2000) and Rabiner (1989). In the case of higher order MEMMs we used beam search, an established approximate inference technique in natural language processing, to infer the most probable sequence (Lowerre, 1976).

3.3. LC-CRF augmented by CS-DCRF

We extend first order LC-CRFs by CS-DCRFs.

Model Definition. First order LC-CRFs model the conditional probability of sequence labels $y_{1:T}$ given a sequence of observed variables $\mathbf{x}_{1:T}$ directly, i.e.

$$p(y_{1:T}|\mathbf{x}_{1:T}) = \frac{\prod_t \phi(y_t, \mathbf{x}_{1:T})\phi(y_{t-1}, y_t)}{Z(\mathbf{x}_{1:T})}, \qquad (12)$$

and

$$Z(\mathbf{x}_{1:T}) = \sum_{y_{1:T}} \prod_t \phi(y_t, \mathbf{x}_{1:T})\phi(y_{t-1}, y_t) \qquad (13)$$

is the partition function.

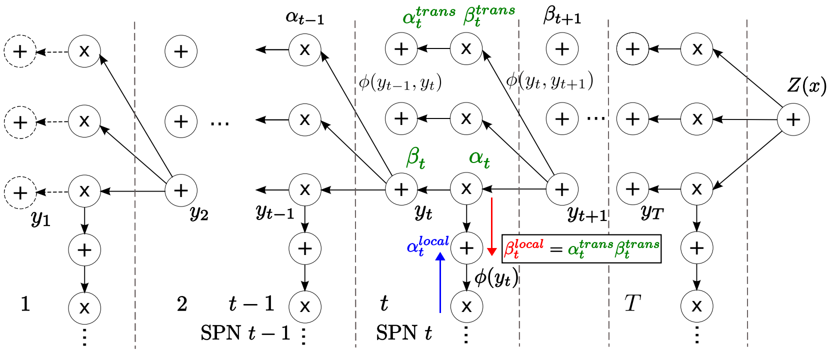

Figure 3. LC-CRF extended by CS-DCRF and its SPN representation illustrating the forward-backward algorithm.

Forward-backward Algorithm on the Chain. We extend the LC-CRF by replacing these local factors $\phi(y_t, \mathbf{x}_{1:T}) = \alpha_t^{local}(y_t, \mathbf{x}_{1:T}) = \phi(y)\alpha^{deep}(y_t, \mathbf{x}_{1:T})$ in Eq. (12) and (13) by CS-DCRFs. Accordingly, we adapt the forward messages

$$\alpha_t^{trans}(y_t) = \sum_{y_{t-1}} \phi(y_{t-1}, y_t)\alpha_{t-1}(y_{t-1}), \tag{14}$$

$$\alpha_t(y_t) = \alpha_t^{local}(y_t, \mathbf{x}_{1:T})\alpha_t^{trans}(y_t) \tag{15}$$

and backward messages

$$\beta_t^{trans}(y_t) = \sum_{y_{t+1}} \phi(y_t, y_{t+1})\beta_{t+1}(y_{t+1}), \tag{16}$$

$$\beta_t(y_t) = \alpha_t^{local}(y_t, \mathbf{x}_{1:T})\beta_t^{trans}(y_t), \tag{17}$$

where $\alpha_t^{trans}(y_t)$ and $\beta_t^{trans}(y_t)$ denote the messages passed along the linear chain without the local message $\alpha_t^{local}(y_t, \mathbf{x}_{1:T})$. Further, $\alpha_t(y_t)$ and $\beta_t(y_t)$ denote the messages passed along the linear chain including the local message at sequence index t. Figure 3 shows a sum-product network representation of the forward-backward algorithm in the linear chain and how it can be extended to deep local factors, i.e. CS-DCRFs.

The backward messages for initializing the backward recursion in the CS-DCRF are

$$\beta_t^{local}(y_t) = \alpha_t^{trans}(y_t)\beta_t^{trans}(y_t) \tag{18}$$

which are needed to compute the backward messages in Table 1 of our CS-DCRF. At this point, the advantage of using the forward-backward algorithm for inference becomes eminent; the forward-backward algorithm can be used for the LC-CRF and the CS-DCRF allowing for joint exact *and* efficient inference and training in a single framework.

Gradients. The gradients for the transition features are

$$\frac{\partial F(\mathbf{w}, \mathcal{D})}{\partial w_{y_{t-1}, y_t}} = \sum_{n=1}^{N}\sum_{t=1}^{T} \delta(y_{t-1,n}, y_{t-1})\delta(y_{t,n}, y_t) - p(y_t,\, y_{t-1}\,|\mathbf{x}_{1:T,n})$$

and the corresponding marginal probabilities are

$$p(y_t,\, y_{t-1}\,|\mathbf{x}_{1:T}) = \frac{\alpha_{t-1}(y_{t-1})\phi(y_{t-1}, y_t)\beta_t(y_t)}{Z(\mathbf{x}_{1:T,n})}.$$

Inference. For details on how to compute the most probable sequence $\hat{y}_{1:T}$

$$\hat{y}_{1:T} = \operatorname*{argmax}_{y_{1:T}} \prod_t \alpha_t^{local}(y_t, \mathbf{x}_{1:T})\phi(y_{t-1}, y_t)$$

using the Viterbi algorithm, we refer to Sutton (2008).

4. Experiments

4.1. Data sets

We evaluated the performance of the proposed models on the following two data sets:

OCR. The OCR data set (Taskar et al., 2004) represents an optical character recognition task. The data set consists of 6877 handwritten words, each represented as a sequence of handwritten characters. These characters are provided as binary images of size 16 × 8 pixels and the raw pixel values serve as input features. The task is to assign one out of 26 possible labels, i.e. the represented character, to each of these images. In total, 55 unique words with an average length of 8 characters are provided. Performance is measured by the ratio of wrong assigned labels to the total number of labels. Furthermore, 10-fold cross-validation is used. Nine parts are used for training and one part for testing. The average accuracy over all ten folds is reported.

TIMIT. The TIMIT data set (Zue et al., 1990) contains recordings of 5.4 hours of English speech from 8 major dialect regions of the United States. The recordings were manually segmented at phone level. We use this segmentation for phone classification. Note that phone classification should not be confused with phone recognition (Hinton et al., 2012) where no segmentation is provided. As suggested in (Lee & Hon, 1989), we collapsed the original 61 phones into 39 phones. For every segment of the recording, we computed 13 Mel frequency cepstral coefficients (MFCCs), delta coefficients and double-delta coefficients as input features. The task is, given an utterance and a corresponding segmentation, to infer the phoneme within every segment. The data set consists of a training set, a development set (dev) and a test set (test), containing 142.910, 15.334 and 7.333 phonetic segments, respectively. Furthermore, the development set is used for parameter tuning.

4.2. Labeling experiments

In all experiments and for all data sets, input features were normalized to zero mean and unit standard deviation. Optimization of our models was in all cases performed using stochastic gradient ascent using a batch-size of one sample. An ℓ_2-norm regularizer on the model weights was used.

Table 2. (OCR task) Comparison of CS-DCRF for different model sizes and its first order extensions (MEMMs and LC-CRFs). Performance measure: Character error rate (CER) in percent.

		$I = 2$		$I = 3$	
L		$H = 2$	$H = 3$	$H = 2$	$H = 3$
1	MEMM	13.87	13.13	12.26	11.81
	LC-CRF	8.35	7.81	7.32	6.94
2	MEMM	10.66	10.40	9.35	9.37
	LC-CRF	6.28	6.53	**5.75**	5.76
3	MEMM	9.37	8.74	9.31	n.a.
	LC-CRF	5.77	6.76	5.87	n.a.

OCR. First, we compared first order MEMMs and LC-CRFs augmented by CS-DCRFs. For inference we used the Viterbi algorithm which is exact and efficient for first order models. In Table 2 we explored the performance of the CS-DCRF extension for various structures, i.e. different number of layers L, number of hidden variables I per layer and number of states H. Increased model size improved clearly the performance. For similar model configurations, the LC-CRFs significantly outperformed the MEMMs.

Second, we considered higher order MEMMs to investigate the influence of longer label history on the performance and present these results in Table 3 for different model sizes using one hidden layer. The longer the history and the larger the model, the better the performance. Furthermore, we explored one to three hidden layers. The best results of the extensive exploration are obtained for $M - 1 = 8$ and shown in Table 4.

Finally, we summarized our best results in Table 5 and compared them. In particular, we compared our models to first order LC-CRFs, first order LC-CRFs augmented by GMMs (special case of our model) and first order hidden-unit CRFs (HU-CRFs) (van der Maaten et al., 2011), optimized using stochastic gradient descent. These models achieved a labeling error of 14.2%, 9.53% ($L = 1$, $I = 1$, $H = 4$) and 7.73% (hidden variables $I = 250$ and states $H = 2$), respectively.[1] All presented models are better than LC-CRFs with linear local factors. Our augmented LC-CRFs achieved better performance (5.70%) than the first order HU-CRFs and first order LC-CRF augmented by GMMs. Our best result (3.12%) has been achieved with the higher order MEMM augmented by a 3-layer CS-DCRF. We expect even better performance for higher order LC-CRFs augmented by CS-DCRFs. However, this is scope of future work.

Table 3. (OCR task) Higher order MEMMs augmented by CS-DCRF for different model sizes using one hidden layer ($L = 1$). Beam search with a width of 20 has been used. Performance measure: Character error rate (CER) in percent.

	$I = 2$		$I = 3$	
$M - 1$	$H = 2$	$H = 3$	$H = 2$	$H = 3$
1	14.32	13.81	12.92	12.39
2	7.73	7.66	7.81	6.98
3	5.64	5.46	5.27	5.17
4	4.49	4.36	4.36	4.17
5	3.39	3.99	3.90	3.80
6	3.78	3.73	3.75	3.57
7	3.73	3.58	3.60	3.36
8	3.69	3.55	3.61	3.37
9	3.68	3.55	3.61	**3.34**

Table 4. (OCR task) Extensive results for higher order MEMMs augmented by CS-DCRF ($M - 1 = 8$). Performance measure: Character error rate (CER) in percent.

	$I = 2$		$I = 3$	
L	$H = 2$	$H = 3$	$H = 2$	$H = 3$
1	3.69	3.55	3.61	3.37
2	3.22	3.45	3.12	3.22
3	**3.12**	3.27	n.a	n.a.

Table 5. (OCR task) Summary. Results marked by (†) are from van der Maaten et al. (2011). Performance measure: Character error rate (CER) in percent.

Model	CER [%]
LC-CRF (1st order)†	14.2
HU-CRF (1st order)†	7.73
GMM+LC-CRF (1st order)	9.53
CS-DCRF+MEMM (1st order)	9.35
CS-DCRF+LC-CRF (1st order)	**5.75**
CS-DCRF+MEMM (higher order)	**3.12**

[1] The state-of-the-art performance on the OCR task was achieved by a second order HU-CRF with large-margin training, achieving an error of 1.99% (van der Maaten et al., 2011).

Table 6. (TIMIT task) Comparison of first order MEMMs and LC-CRFs augmented by CS-DCRF for various model sizes. Performance measure: Phone error rate (PER) in percent.

CS-DCRF		$I = 2$			$I = 3$			$I = 4$		
+MEMM		$H = 2$	$H = 3$	$H = 4$	$H = 2$	$H = 3$	$H = 4$	$H = 2$	$H = 3$	$H = 4$
1 layer	dev	23.04	22.30	21.88	22.35	21.68	22.52	22.21	21.44	21.30
	test	23.58	22.97	22.70	23.30	22.60	22.25	23.20	22.26	22.40
2 layer	dev	21.58	21.11	21.06	21.01	**20.55**	20.60	21.29	20.84	20.62
	test	22.18	22.13	21.97	22.02	**22.15**	21.55	22.78	22.22	21.96
+LC-CRF										
1 layer	dev	21.92	21.21	21.03	21.20	20.77	20.69	21.21	20.52	20.29
	test	22.40	22.00	21.90	21.96	21.52	21.03	21.71	20.92	20.73
2 layer	dev	20.51	20.24	20.23	20.23	19.92	**19.88**	20.37	20.05	19.89
	test	21.08	20.98	20.74	21.14	21.17	**20.90**	21.75	21.66	20.93

TIMIT. Detailed results for our augmented MEMMs and LC-CRFs on the development set as well as the core-test set, including various structures of the CS-DCRF, are provided in Table 6. Augmented LC-CRFs outperformed augmented MEMMs. Larger model sizes improved the performance. We achieved our best performance of 19.95% (not shown in Table 6) with the augmented LC-CRF (L=1, I=6, H=7) using three segments as input, i.e. the input features of neighboring segments are added.

Table 7. (TIMIT task) Summary of labeling results. Results marked by (†) are from (Sung et al., 2007), by (††) are from (Sha & Saul, 2006) and by (†††) are from (Halberstadt & Glass, 1997). Performance measure: Phone error rate (PER) in percent.

Model	PER [%]
GMMs ML††	25.9
HCRFs†	21.5
Large-Margin GMM††	21.1
Heterogeneous Measurements†††	21.0
CNF	20.67
GMM+LC-CRF; 1 seg.	22.72
GMM+LC-CRF diag; 1 seg.	24.21
GMM+LC-CRF; 3 seg.	22.10
CS-DCRF+MEMM	22.15
CS-DCRF+LC-CRF	**20.54**
CS-DCRF+LC-CRF; 3 seg.	**19.95**

Finally, we summarized our results in Table 7 and compared to other state-of-the-art methods, namely hidden conditional random fields (HCRFs) (Sung et al., 2007), large-margin GMMs (Sha & Saul, 2006), heterogeneous measurements (Halberstadt & Glass, 1997) and CNFs (Peng et al., 2009). Using the software of Peng et al. (2009) we tested CNFs with 50, 100 and 200 gates as well as one and three input segments. We achieved the best result with 100 gates and one segment. Large-margin GMMs outperformed generative GMMs and LC-CRFs augmented by GMMs. However, our best model, the LC-CRF augmented by CS-DCRF, outperformed the other state-of-the-art methods.

5. Discussion and Future Work

We considered context-specific deep CRFs (CS-DCRFs) based on sum-product networks enabling both exact and efficient inference. Furthermore, we extended linear-chain CRFs and maximum entropy Markov models (MEMMs) by replacing local factors with CS-DCRFs. Additionally, we formulated the forward-backward algorithm for joint training of the deep model and the linear-chain CRF as well as the MEMM. Finally, we empirically evaluated our models for sequence labeling. Results for phone classification and optical character recognition are provided and are competitive in all cases.

In future work, we will investigate higher order LC-CRFs of the proposed model and include a margin-based objective. Further, we will perform extensive experiments on big data.

6. ACKNOWLEDGMENTS

This work was supported by the Austrian Science Fund (FWF) under the project number P25244-N15.

References

A. L. Berger, V. J. D. Pietra, and S. A. D. Pietra. A maximum entropy approach to natural language processing. *Comput. Linguist.*, 22(1):39–71, March 1996.

C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in bayesian networks. In E. Horvitz and F. V. Jensen (eds.), *UAI*, pp. 115–123. Morgan Kaufmann, 1996. ISBN 1-55860-412-X.

A. Darwiche. A differential approach to inference in bayesian networks. In *Journal of the ACM*, pp. 123–132, 2000.

A. Darwiche. A differential approach to inference in bayesian networks. *J. ACM*, 50(3):280–305, May 2003.

T. M. T. Do and T. Artières. Neural conditional random fields. *Journal of Machine Learning Research - Proceedings Track*, 9:177–184, 2010.

E. Fosler-Lussier, Y. He, P. Jyothi, and R. Prabhavalkar. Conditional random fields in speech, audio, and language processing. *Proceedings of the IEEE*, 101(5): 1054–1075, May 2013.

N. Friedman, D. Geiger, M. Goldszmidt, G. Provan, P. Langley, and P. Smyth. Bayesian network classifiers. In *Machine Learning*, pp. 131–163, 1997.

R. Gens and P. Domingos. Discriminative learning of sum-product networks. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 3248–3256, 2012.

A. Gunawardana, M. Mahajan, A. Acero, and J. C. Platt. Hidden conditional random fields for phone classification. In *in Interspeech*, pp. 1117–1120, 2005.

A. K. Halberstadt and J. R. Glass. Heterogeneous acoustic measurements for phonetic classification. In *EUROSPEECH*, pp. 401–404, 1997.

G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition. *Signal Processing Magazine*, 29(6):82–97, 2012.

G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8): 1771–1800, 2002.

A. Kulesza and F. Pereira. Structured learning with approximate inference. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis (eds.), *NIPS*. Curran Associates, Inc., 2007.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning (ICML)*, pp. 282–289, 2001.

H. Larochelle and Y. Bengio. Classification using discriminative restricted Boltzmann machines. In *International Conference on Machine Learning (ICML)*, pp. 536–543, 2008.

K.-F. Lee and H.-W. Hon. Speaker-independent phone recognition using hidden markov models. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 37(11):1641–1648, 1989.

B. T. Lowerre. *The Harpy Speech Recognition System.* PhD thesis, Pittsburgh, PA, USA, 1976. AAI7619331.

A. McCallum, D. Freitag, and F. C. N. Pereira. Maximum entropy Markov models for information extraction and segmentation. In *International Conference on Machine Learning (ICML)*, pp. 591–598, 2000.

H. Nyman, J. Pensar, T. Koski, and J. Corander. Stratified graphical models - context-specific independence in graphical models. 2013.

J. Peng, L. Bo, and J. Xu. Conditional neural fields. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 1419–1427, 2009.

H. Poon and P. Domingos. Sum-product networks: A new deep architecture. In *Uncertainty in Artificial Intelligence (UAI)*, pp. 337–346, 2011.

R. Prabhavalkar and E. Fosler-Lussier. Backpropagation training for multilayer conditional random field based phone recognition. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pp. 5534–5537, 2010.

L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pp. 257–286, 1989.

F. Sha and L. Saul. Large margin Gaussian mixture modeling for phonetic classification and recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 265–268, 2006.

Y.-H. Sung, C. Boulis, C. Manning, and D. Jurafsky. Regularization, adaptation, and non-independent features improve hidden conditional random fields for phone classification. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pp. 347–352, 2007.

C. Sutton. *Efficient Training Methods for Conditional Random Fields*. PhD thesis, University of Massachusetts, 2008.

D. Tarlow, I. E. Givoni, and R. S. Zemel. Hop-map: Efficient message passing with high order potentials. In *Proceedings of 13th Conference on Artificial Intelligence and Statistics*, 2010.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2004.

L. van der Maaten, M. Welling, and L. K. Saul. Hidden-unit conditional random fields. *Journal of Machine Learning Research - Proceedings Track*, 15:479–488, 2011.

V. Zue, S. Seneff, and J. R. Glass. Speech database development at MIT: Timit and beyond. *Speech Communication*, 9(4):351–356, 1990.