# STRUCTURED PREDICTION ENERGY NETWORKS

**David Belanger & Andrew McCallum**
School of Information and Computer Science
University of Massachusetts Amherst
{belanger,mccallum}@cs.umass.edu

## ABSTRACT

We introduce *structured prediction energy networks* (SPENs), a flexible framework for structured prediction. A deep architecture is used to define an *energy function* of candidate labels, and then predictions are produced by using backpropagation to iteratively optimize the energy with respect to the labels. This deep architecture captures dependencies between labels that would lead to intractable graphical models, and performs *structure learning* by automatically learning discriminative features of the structured output. One natural application of our technique is multi-label classification, which traditionally has required strict prior assumptions about the interactions between labels to ensure tractable learning and prediction problems. We are able to apply SPENs to multi-label problems with substantially larger label sets than previous applications of structured prediction, while modeling high-order interactions using minimal structural assumptions. Overall, deep learning provides remarkable tools for learning features of the inputs to a prediction problem, and this work extends these techniques to learning features of the outputs. Our experiments provide impressive performance on a variety of benchmark multi-label classification tasks, demonstrate that our technique can be used to provide interpretable structure learning, and illuminate fundamental trade-offs between feed-forward and iterative structured prediction techniques.

## 1 INTRODUCTION

Structured prediction is an important problem in a variety of machine learning domains. Consider an input $x$ and structured output $y$, such as a labeling of time steps, a collection of attributes for an image, a parse of a sentence, or a segmentation of an image into objects. Such problems are challenging because the number of candidate $y$ is exponential in number of output variables that comprise it. As a result, practitioners encounter *computational* considerations, since prediction requires searching the enormous space of outputs, and also *statistical* considerations, since learning accurate models from limited data requires reasoning about commonalities between distinct structured outputs. Therefore, structured prediction is fundamentally a problem of representation, where the representation must capture both the discriminative interactions between $x$ and $y$ and also allow for efficient combinatorial optimization over $y$. With this perspective in mind, it is not surprising that there are variety of natural ways to couple structured prediction with deep learning, a powerful framework for representation learning.

We consider two principal approaches to structured prediction: (a) as a *feed-forward* function $y = f(x)$, and (b) using an *energy-based* viewpoint $y = \arg\min_{y'} E_x(y')$ (LeCun et al., 2006). Feed-forward approaches include, for example, predictors using local convolutions plus a classification layer (Collobert et al., 2011), fully-convolutional networks (Long et al., 2015), or sequence-to-sequence predictors (Vinyals et al., 2014). In contrast, the energy-based approach may involve non-trivial optimization to perform predictions, and includes, for example, conditional random fields (CRFs) (Lafferty et al., 2001). From a modeling perspective, energy-based approaches are desirable because directly parametrizing $E_x()$ provides practitioners with better opportunities to utilize domain knowledge about properties and invariances of the structured output. Furthermore, such a parametrization may be more parsimonious, resulting in improved generalization from limited data. On the other hand, for energy-based models, both prediction and learning are more complex than for feed-forward approaches.

Both approaches can be combined naturally with deep learning. It is straightforward to parametrize a feed-forward predictor $y = f(x)$ as a deep architecture. In these, end-to-end learning can be performed easily using gradient descent. For energy-based prediction, prior applications of deep learning have mostly followed a two-step construction: first, choose an existing model structure for which the search problem $y = \arg\min_{y'} E_x(y')$ can be performed efficiently, eg. with the Viterbi algorithm, and then express the dependence of $E_x()$ on $x$ via a deep architecture. For example, in a CRF, the tables of potentials of an undirected graphical model can be parametrized via a deep network applied to $x$ (LeCun et al., 2006; Collobert et al., 2011; Huang et al., 2015). The advantage of this approach is that it leverages the strength of deep architectures to perform representation learning on $x$, while maintaining the ability to perform efficient combinatorial prediction, since the dependence of $E_x(y')$ on $y'$ remains unchanged. However, by assuming a particular graphical model structure for $E_x()$ a-priori, this construction perhaps imposes an excessively strict inductive bias and, and practitioners are unable to use the deep architecture to perform *structure learning*, ie. representation learning that discovers the interaction between different parts of $y$.

In response, we present *structured prediction energy networks* (SPENs), a novel energy-based prediction technique that offers substantially different tradeoffs than these prior applications of deep learning to structured prediction. Namely, we sacrifice algorithmic guarantees for solving $\arg\min_{y'} E_x(y')$ exactly, in exchange for an extremely flexible framework for expressing the energy function $E_x()$. We use a deep architecture to encode the energy , and perform predictions by approximately minimizing the energy with respect to the prediction variables $y$ using gradient descent, where gradients are obtained by backpropagation through the deep architecture. The parameters of the network are trained using an adaptation of a structured SVM (Taskar et al., 2004; Tsochantaridis et al., 2004). The deep network allows us to model high-arity interactions that would result in unmanageable treewidth if the problem was posed as an undirected graphical model. Furthermore, the learned *measurement matrix* (Section 3) of the SPEN provides an interpretable tool for structure learning.

Typically, back-propagation through a deep architecture is used during learning to update the network parameters. However, there is a breadth of work, both old and contemporary, on using back-propagation to update prediction variables (Bromley et al., 1993; Szegedy et al., 2014; Goodfellow et al., 2014; Le & Mikolov, 2014; Mordvintsev et al., 2015; Gatys et al., 2015a;b). As with this prior work, prediction in SPENs is conceptually simple and easy to implement.

Our experiments focus on applying SPENs to multi-label classification problems. These are naturally posed as structured prediction problems, since the labels exhibit rich interaction structure. However, prior applications of structured prediction, eg. using CRFs, have been limited to notably smaller problems than our experiments consider, since the techniques' computational complexity either grows super-linearly in the number of labels $L$ (Ghamrawi & McCallum, 2005; Finley & Joachims, 2008; Meshi et al., 2010; Petterson & Caetano, 2011), or requires strict assumptions about the dependencies between labels (Read et al., 2011; Jasinska & Dembczyski, 2015; Niculescu-Mizil & Abbasnejad, 2015). SPENs, on the other hand, scale linearly in $L$ while placing mild prior assumptions about the labels' interactions, namely that they can be encoded by a deep architecture.

On a variety of benchmark multi-label classification tasks, the expressivity of our deep energy function provides accuracy improvements against a variety of competitive baselines. We also offer experiments on synthethic data with rigid mutual exclusivity constraints between labels to demonstrate the power of SPENs to perform structure learning. These experiments illuminate important tradeoffs in the expressivity and parsimony of SPENs vs. feed-forward predictors. We encourage future work using energy-based structured prediction in deep learning.

## 2 Structured Prediction Energy Networks

A fully general way to specify the set of all $x \rightarrow y$ mappings is to pose $y$ as the solution to a potentially non-linear combinatorial optimization problem, with parameters dependent on $x$:

$$\min_{y} \quad \mathrm{E_x}(y) \quad \text{subject to} \qquad y \in \{0,1\}^L. \tag{1}$$

The structured prediction problem (1) could be rendered tractable by assuming certain specific structure for the energy function $E_x()$, such as a tree-structured undirected graphical model. Instead, we consider general $E_x()$, but optimize over a convex relaxation of the constraint set:

$$\min_{y} \quad \mathrm{E_x}(\bar{y}) \quad \text{subject to} \qquad \bar{y} \in [0,1]^L. \tag{2}$$

In general, $E_x(\bar{y})$ may be non-convex, so exactly solving (2) may be intractable. A reasonable approximate optimization procedure, however, is to minimize (2) via gradient descent, obtaining a local minimum. Optimization over the set $[0,1]^L$ can be performed using entropic mirror descent (aka exponentiated gradient) by normalizing over each coordinate (Beck & Teboulle, 2003).

There are no guarantees that our predicted $\bar{y}$ values are nearly 0-1. In some applications, we may need to round $\bar{y}$ to obtain predictions that are usable downtream. Sometimes, it is useful to maintain 'soft' predictions, eg. for detection problems, since we may want to threshold based on confidence.

In the posterior inference literature, *mean-field* approaches also consider a relaxation from $y$ to $\bar{y}$, where $\bar{y}_i$ would be interpreted as the marginal probability that $y_i = 1$ (Jordan et al., 1999). Here, the practitioner starts with a probabilistic model for which inference is intractable, and obtains a mean-field objective when seeking to perform approximate variational inference. We make no such probabilistic assumptions, however, and instead adopt a disciminative approach by directly parametrizing the objective that the inference procedure optimizes.

Continuous optimization over $\bar{y}$ can be performed using black-box access to a gradient subroutine for $E_x(\bar{y})$. Therefore, it is natural to parametrize $E_x(\bar{y})$ using deep architectures, a flexible family of multivariate function approximators that provide efficient gradient calculation.

A SPEN parameterizes $E_x(\bar{y})$ as a neural network that takes both $x$ and $\bar{y}$ as inputs and returns the energy (a single number). In general, a SPEN consists of two deep architectures. First, the *feature network* F($x$) produces an $f$-dimensional feature representation for the input. Next, the energy $E_x(\bar{y})$ is given by the output of the *energy network* $G(F(x), \bar{y})$. Here, $F$ and $G$ can be arbitrary deep networks.

Note that the energy only depends on $x$ via the value of $F(x)$. During iterative prediction, we improve efficiency by precomputing $F(x)$ and not back-propagating through $F$ when differentiating the energy with respect to $\bar{y}$.

## 3 Example SPEN Architecture

We now provide a more concrete example of the architecture for a SPEN. All of our experiments use the general configuration described in this section. We denote matrices in upper case and vectors in lower case. We use $g()$ to denote a coordinate-wise non-linearity function, and may use different non-linearities, eg. sigmoid vs. rectifier, in different places.

For our feature network, we employ a simple 2-layer neural network:

$$f(x) = g(A_2 g(A_1 x)). \tag{3}$$

Our energy network is the sum of two terms. First, the *local energy network* scores $\bar{y}$ as the sum of $L$ independent linear models:

$$E_x^{\text{local}}(\bar{y}) = \sum_{i=1}^{L} \bar{y}_i b_i^\top f(x). \tag{4}$$

Here, each $b_i$ is an $F$ dimensional vector of parameters for every label.

This score is added to the output of the *label energy network*, which scores configurations of $\bar{y}$ independent of $x$:

$$E_x^{\text{label}}(\bar{y}) = c_2^\top g(C_1 \bar{y}). \tag{5}$$

The product $C_1 \bar{y}$ is a set of learned linear measurements of the output, that capture salient features of the labels used to model their dependencies. By learning the *measurement matrix* $C_1$ from data, the practitioner imposes minimal assumptions a-priori on the interaction structure between the labels.

While computing such a product is only linear in $L$, we can model sophisticated interactions by feeding $C_1\bar{y}$ through a non-linear energy. In Section 7.2, we present experiments exploring the usefulness of the measurement matrix as a means to perform structure learning.

In some of our experiments, we add another layer of depth to (5). In general, there is a tradeoff between using increasingly expressive energy networks and being more vulnerable to overfitting.

In future work, it would be natural to use a label energy network that conditions on $x$. For example,

$$E_x^{\text{cond}}(\bar{y}) = d_2^\top g(D_1[\bar{y}; f(x)]). \tag{6}$$

IAlso, it may be desirable to choose $g$ that result in a convex prediction problem. However, our experiments select $g$ based on accuracy, rather than algorithmic guarantees resulting from convexity.

### 3.1 Conditional Random Fields as SPENs

There are important parallels between the example SPEN architecture given above and the parametrization of a CRF (Lafferty et al., 2001; Sutton & McCallum, 2011). Here, we use CRF to refer to any structured linear model, which may or may not be trained to maximize the conditional log likelihood. For the sake of notational simplicity, consider a fully-connected pairwise CRF with local potentials that depend on $x$, but data-independent pairwise potentials. Let vec() flatten a matrix into a vector. Suppose we apply $E_x()$ directly to $y$, rather than to the relaxation $\bar{y}$. The corresponding label energy net would be:

$$E_x^{\text{crf}}(y) = s_2^\top \text{vec}(yy^\top), \tag{7}$$

In applications with large label spaces, (7) is troublesome in terms of both the statistical efficiency of parameter estimation and the computational efficiency of prediction because of the quadratic dependence on $L$. Statistical issues can be mitigated by imposing parameter tying of the CRF potentials, using a low-rank assumption, eg. (Srikumar & Manning, 2014; Jernite et al., 2015), or using a deep architecture to map $x$ to a table of CRF potentials (LeCun et al., 2006). Computational concerns can be mitigated by choosing a sparse graph. This is difficult for practitioners when they do not know the dependencies between labels a-priori. Furthermore, modeling high-order interactions than pairwise relationships is very expensive with a CRF, but presents no extra cost for SPENs.

For CRFs, the interplay between the graph structure and the set of representable conditional distributions is well-understood (Koller & Friedman, 2009). However, characterizing the representational capacity of SPENs is more complex, as it depends on the general representational capacity of the deep architecture chosen.

## 4 Learning SPENs

In Section 2, we described a technique for producing predictions by performing continuous optimization in the space of outputs. Now, we discuss a gradient-based technique for learning the parameters of the deep architecture $E_x(\bar{y})$.

In many structured prediction applications, the practitioner is able to interact with the model in only two ways: (1) evaluate the model's energy on a given value of $y$, and (2) minimize the energy with respect to the $y$. This occurs, for example, when predicting combinatorial structures such as bipartite matchings and graph cuts. A popular technique in these settings is the structured support vector machine (SSVM) (Taskar et al., 2004; Tsochantaridis et al., 2004).

If we assume (incorrectly) that our prediction procedure is not subject to optimization errors, then (1) and (2) apply to our model and it is straightforward to train using an SSVM. This ignores errors resulting from the potential non-convexity of $E_x(\bar{y})$ or the relaxation from $y$ to $\bar{y}$. However, such an assumption is a reasonable way to construct an approximate learning procedure.

Define $\Delta(y_p, y_g)$ to be an error function between a prediction $y_p$ and the ground truth $y_g$, such as the Hamming loss. Let $\Psi$ denote the parameters of $E_x$. Let $[\cdot]_+ = \max(0, \cdot)$. The SSVM minimizes the training objective

$$L(\Psi) = \sum_{\{x_i, y_i\}} \max_y \left[\Delta(y_i, y) - E_{x_i}(y) + E_{x_i}(y_i)\right]_+ . \tag{8}$$

Note that the signs in (8) differ from convention because here prediction minimizes $E_x()$. We minimize our loss with respect to the parameters of the deep architecture $E_x$ using mini-batch stochastic gradient descent. For a given $\{x_i, y_i\}$, the subgradient of 8 is:

$$\nabla_\Psi L(\Psi) = I\left[\Delta(y_i, y_\text{p}) - E_{x_i}(y_\text{p}) + E_{x_i}(y_i) > 0\right]\left(-\nabla_\Psi E_{x_i}(y_\text{p}) + \nabla_\Psi E_{x_i}(y_i)\right) \tag{9}$$

Here, $I[\cdot]$ is an indicator function for a predicate, and $y_\text{p}$ is the output of *loss-augmented inference*:

$$y_p = \arg\min_y \left(-\Delta(y_i, y) + E_{x_i}(y)\right). \tag{10}$$

With this, (9) can be computed using back-propagation through $E_x$.

We perform loss-augmented inference by again using gradient descent on the relaxation $\bar{y}$, rather than performing combinatorial optimization over $y$. Since $\Delta$ is a discrete function such as the Hamming loss, we need to approximate it with a differentiable surrogate, such as the squared loss. Any surrogate loss used for training a feed-forward predictor with gradient descent can be used here. Note that the objective (8) only considers the energy values of the ground truth and the prediction, ensuring that they're separated by a margin, not the actual ground truth and predicted labels (10). Therefore, we do not round the output of (10) in order to approximate a subgradient of (8); instead, we evaluate the energy directly on the $\bar{y}$ obtained by approximately minimizing (10).

Finally, we have found that it is useful to initialize the parameters of the feature network by first training them using a simple local classification loss, ignoring any interactions between coordinates of $y$. For problems with very limited training data, we have found that overfitting can be lessened by keeping the feature network's parameters fixed when training the label energy network parameters.

# 5 Applications of SPENs

Our experiments focus on multi-label classification, an important task in a variety of machine learning applications. The data consist of $\{x, y\}$ pairs, where $y = \{y_1, \ldots, y_L\} \in \{0,1\}^L$ is a set of multiple binary labels we seek to predict and $x$ is a feature vector. In many cases, we are given no structure among the $L$ labels a-priori, though the labels may be quite correlated. SPENs are a very natural model for multi-label classification because learning the measurement matrix $C_1$ in (5) provides an automatic method for discovering this interaction structure. Section 6.3 discusses the relationship between SPENs and prior work on multi-label classification.

SPENs are very general, though, and can be applied to any prediction problem that can be posed as MAP inference in an undirected graphical model. In many applications of graphical models, the practioner employs certain prior knowledge about dependencies in the data to choose the graph structure, and certain invariances in the data to impose parameter tying schemes. For example, when tagging sequences with a linear-chain CRF, the parameterization of local and pairwise potential functions is shared across time. Similarly, when applying a SPEN, we can express the label energy net (5) using temporal convolutions, ie. $C_1$ has a repeated block-diagonal structure.

Section A describes details for improving the accuracy and efficiency of SPENs in practice.

# 6 Related Work

## 6.1 Iterative Prediction using Neural Networks

Our use of backprogation to perform gradient-based prediction differs from most deep learning applications, where backpropagation is used to update the network parameters. However, backpropagation-based prediction has been useful in a variety of deep learning applications, including *siamese networks* (Bromley et al., 1993), methods for generating adversarial examples (Szegedy et al., 2014; Goodfellow et al., 2014), methods for embedding documents as dense vectors (Le & Mikolov, 2014), and successful techniques for image generation and texture synthesis (Mordvintsev et al., 2015; Gatys et al., 2015a;b).

In concurrent work, (Carreira et al., 2015) propose an iterative structured prediction method for human pose estimation, where $E_x(y)$, doesn't return a number, but instead an increment $\Delta(x, y)$. Predictions are constructed by incrementally stepping as $y_{t+1} = y_t + \Delta(x, y_t)$. The $\Delta$ network is trained as a multi-variate regression task, by defining a ground truth trajectory for intermediate $y_t$.

An alternative line of work has constructed feed-forward predictors using energy-based models as motivation (Domke, 2013; Hershey et al., 2014; Zheng et al., 2015). Here, an energy-based model family is chosen, along with an iterative inference technique for the model. The inference technique is then unrolled into a computation graph, for a fixed number of iterations, and all of the parameters are learned end-to-end by back-propagating through the iterative procedure. Such an approach presents different inductive biases than our approach, since it introduces many additional parameters, but places rigid restrictions on how they are used.

### 6.2 Conditional Random Fields

A natural alternative to SPENs for structured prediction is to encode $E_x(y)$ as a CRF. the principal advantage of SPENs is that CRF inference is exponential in the treewidth of the graph, whereas the measurements employed by SPENs can extract information from arbitrarily many labels at once. While the per-iteration complexity of SPEN prediction is superior to CRFs of comparable expressivity, it is difficult to analyze its overall cost compared to CRFs, eg. with belief propagation, because both perform non-convex optimization.

Training CRFs using an SSVM loss is conceptually more attractive than training SPENs, however. In loopy graphical models, it is tractable to solve the LP relaxation of MAP inference using graph-cuts or message passing techniques, eg. (Boykov & Kolmogorov, 2004; Globerson & Jaakkola, 2008). Solving the LP relaxation, instead of performing exact MAP inference, in the inner loop of SSVM learning is fairly benign, since it is guaranteed to over-generate margin violations in (8). A chief concern when training a SPEN with an SSVM is that the non-convex optimization in the inner loop of learning will find poor local minima such that no margin violations in (8) are discovered (Kulesza & Pereira, 2007; Finley & Joachims, 2008). Since parameter updates (9) only occur when margin violations are discovered, this halts the learning process.

### 6.3 Multi-Label Classification

The most simple multi-label classification approach is to independently predict each label $y_i$ using a separate classifier, also known as the 'binary relevance model' (Tsoumakas & Katakis, 2006). This can perform poorly, particularly when certain labels are rare or some are highly correlated. Modeling improvements use max-margin or ranking losses that directly address the multi-label structure (Elisseeff & Weston, 2001; Godbole & Sarawagi, 2004; Zhang & Zhou, 2006; Bucak et al., 2009).

Correlations between labels can be modeled explicitly using models with low-dimensional embeddings of labels (Ji & Ye, 2009; Cabral et al., 2011; Yu et al., 2014; Xu et al., 2014; Bhatia et al., 2015). This can be achieved, for example, by using low-rank parameter matrices. In the SPEN framework, such a model would consist of a linear feature network (3) of the form $f(x) = A_1 x$, where $A_1$ has fewer rows than there are target labels, and no label energy network. While the prediction cost of such methods grows linearly with $L$, these models have limited expressivity, and can not capture strict structural constraints among labels, such as mutual exclusivity and implicature. By using a non-linear multi-layer perceptron (MLP) for the feature network with hidden layers of lower dimensionality than the input, we are able to capture similar low-dimensional structure, but also capture interactions between outputs. In our experiments, an MLP is a novel, competitive baseline.

It is natural to approach multi-label classification using structured prediction, which models interactions between prediction labels directly. However, these techniques' computational complexity grows super-linearly in $L$ (Ghamrawi & McCallum, 2005; Finley & Joachims, 2008; Meshi et al., 2010; Petterson & Caetano, 2011) , or requires practitioners to impose strict assumption about the dependencies between labels (Read et al., 2011; Jasinska & Dembczyski, 2015; Niculescu-Mizil & Abbasnejad, 2015). This has prevented scalability to large label spaces with complex interactions.

Our parametrization of the label energy network (5) in terms of linear measurements of the labels is inspired by prior approaches using compressed sensing and error-correcting codes for multi-label classification (Hsu et al., 2009; Hariharan et al., 2010; Kapoor et al., 2012). However, these rely on assumptions about the sparsity of the true labels or prior knowledge about label interactions, and often do not learn the measurement matrix from data. We do not assume that the labels are sparse. Instead, we assume that their interaction can be parametrized by a deep network applied to a set of linear measurements of the labels.

| Method | BR | LR | MLP | SPEN |
|---|---|---|---|---|
| Bibtex | 37.2 | 39.0 | 38.9 | **42.2** |
| Delicious | 26.5 | 35.3 | **37.0** | 35.2 |
| Bookmarks | 30.7 | 31.0 | 33.8 | **34.4** |

Table 1: Comparison of various methods on 3 standard datasets in terms of F1 (larger is better).

# 7 Experiments

## 7.1 Multi-Label Classification Benchmarks

Table 1 compares SPENs to a variety of high-performing baselines on a selection of standard multi-label classification tasks (Tsoumakas & Katakis, 2006). Dataset sizes, etc. are described in Table 4. We compare **BR**: independent per-label logistic regression, ie. the 'binary relevance model' Tsoumakas & Katakis (2006). **MLP**: multi-layer perceptron with ReLU non-linearities trained with per-label logistic loss, ie. the feature network equation (3) coupled with the local energy network equation (4). **LR**: the low-rank-weights method of Yu et al. (2014). All results besides **MLP** and **SPEN**, are taken from Lin et al. (2014). We report the 'example averaged' F1 measure. For Bibtex and Delicious, we tune parameters by first jack-knifing a separate train-test split. For Bookmarks, we use the same train-dev-test split as Lin et al. (2014). For SPENs, we obtain predictions by rounding $\bar{y}_i$ above a threshold tuned on held-out data. Section A.2 describes our hyperparameters.

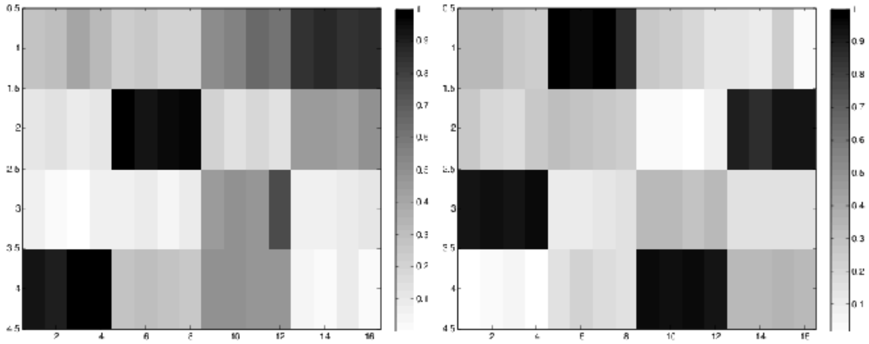
There are multiple key results in Table 1. First, SPENs are very competitive compared to all of the other methods. Second, MLP, a technique that has not been treated as a baseline in recent literature, is surprisingly accurate as well. Finally, the MLP outperformed SPEN on the Delicious dataset. Here, we found that accurate prediction requires well-calibrated soft predictions to be combined with a confidence threshold. The MLP, which is trained with logistic regression, is better at predicting soft predictions than SPENs, which are trained with a margin loss. To obtain the SPEN result for Delicious in Table 1, we need to smooth the test-time prediction problem with extra entropy terms to obtain softer predictions.

Many multi-label classification methods approach learning as a missing data problem. Here, the training labels $y$ are assumed to be correct only when $y_i = 1$. When $y_i = 0$, they are treated as missing data, whose values can be imputed using assumptions about the rank (Lin et al., 2014) or sparsity (Bucak et al., 2011; Agrawal et al., 2013) of the matrix of training labels. For certain multi-label tasks, such modeling is useful because only positive labels are annotated. For example, the approach of (Lin et al., 2014) achieves 44.2 on the Bibtex dataset, outperforming our method, but only 33.3 on Delicious, substantially worse than the MLP or SPEN. Missing data modeling is orthogonal to the modeling of SPENs, and we can combine missing data techniques with SPENs.

Generally, SPENs are slower than non-iterative prediction techniques. However, for some problems the evaluation of the feature network (3) is a substantial burden, and this only needs be run once, for SPENs since the features do not depend on $y$. We also perform predictions across very large batches on a GPU in parallel. Despite providing substantial speedups, this approach is subject to the 'curse of the last reducer,' where unnecessary gradient computation is performed on easy examples while waiting for difficult examples to converge. Finally, the speed and accuracy of SPENs can be traded off by modifying the convergence criterion, learning rate, etc. of the prediction-time optimization.

## 7.2 Performing Structure Learning Using SPENs

Next, we perform experiments on synthetic data designed to demonstrate that the label measurement matrix, $C_1$ in the label energy network (5), provides a useful tool for analyzing the structure of dependencies between labels. SPENs impose a particular inductive bias about the interaction between $x$ and $y$. Namely, the interactions between different labels in $y$ do not depend on $x$. Our experiments show that this parametrization allows SPENs to excel in regimes of limited training data, due to their superior parsimony compared to analogous feed-forward approaches.



(a) ReLU (b) HardTanh

Figure 1: Learned SPEN measurement matrices on synthetic data containing mutual exclusivity of labels within size-4 blocks, for two different choices of nonlinearity in the label energy network. 16 Labels on horizontal axis and 4 hidden units on vertical axis.

To generate data, we first draw a design matrix $X$ with 64 features, with each entry drawn from $N(0,1)$. Then, we generate a 64 x 16 weights matrix $A$, again with entries from $N(0,1)$. Then, we construct $Z = XA$ and split the 16 columns of $Z$ into 4 consecutive blocks. For each block, we set $Y_{ij} = 1$ if $Z_{ij}$ is the maximum entry in its row-wise block, and 0 otherwise. We seek to learn a model with predictions that reliably obey these within-block mutual exclusivity constraints.

Figure 1 depicts block structure in the learned measurement matrix. Measurements that place equal weight on every element in a block can be used to detect violations of the mutual exclusivity constraints characteristic of the data generatic process. The choice of network architecture can significantly affect the interpretability of the measurement matrix, however. When using ReLU, which acts as the identity for positive activations, violations of the data constraints can be detected by taking linear combinations of the measurements (a), since multiple hidden units place large weight on some labels. This obfuscates our ability to perform structure learning by investigating the measurement matrix. On the other hand, since applying HardTanh to measurements saturates from above, the network learns to utilize each measurement individually, yielding substantially more interpretable structure learning in (b).

Next, in Table 2 we compare: a linear classifier, a 3-Layer ReLU MLP with hidden units of size 64 and 16, and a SPEN with a simple linear local energy network and a 2-layer label energy network with HardTanh activations and 4 hidden units. Using fewer hidden units in the MLP results in substantially poorer performance. We avoid using a non-linear local energy network in the SPEN because we want to force the label energy network to capture all interactions between labels.

Note that the SPEN consistently outperforms the MLP, particularly when training on only 1.5k examples. In the limited data regime, their difference is because the MLP has 5x more parameters, since we use a simple linear feature network in the SPEN. We also because we inject domain knowledge about the constraint structure when designing the label energy network's architecture. Figure 2 in the Appendix demonstrates that we can perform the same structure learning as in Figure 1 on this small training data.

Next, observe that for 15k examples the performance of the MLP and SPEN are comparable. Initially, we hypothesized that the mutual exclusivity constraints of the labels could not be satisfied by a feed-forward predictor, and that reconciling their interactions would require an iterative procedure. However, it seems that a large, expressive MLP can learn an accurate predictor when presented with lots of examples. Going forward, we would like to investigate the parsimony vs. expressivity tradeoffs of SPENs and MLPs.

### 7.3 Analyzing the Effect of Search Errors on SSVM Training

Due to scalability considerations, prior applications of CRFs to multi-label classification have been restricted to substantially smaller $L$ than those considered in Table 1. In Table 3, we consider

| # train examples | Linear | 3-Layer MLP | SPEN w/ Linear Local Energy |
|---|---|---|---|
| 1.5k | 80.0 | 81.6 | **91.5** |
| 15k | 81.8 | 96.3 | **96.7** |

Table 2: Comparing F1 performance on the synthetic task with block-strucutred mutual exclusivity between labels. Due to its parsimonious parametrization, the SPEN succeeds with limited data. With more data, the MLP performs comparably to the SPEN, suggesting that even rigid constraints among labels can be predicted in a feed-forward fashion using a sufficiently expressive architecture.

| GREEDY | LBP | EXACT | LP | SPEN |
|---|---|---|---|---|
| 21.6 ± .56 | 24.3 ± .61 | 20.23 ± .53 | 20.49 ± .54 | 20.88 ± .19 |

Table 3: Comparing different prediction methods, which are used both during SSVM training and at test time, using the setup of Finley & Joachims (2008) on the Yeast dataset. We report hamming error (smaller is better). SPENs perform comparably to EXACT and LP, which provide stronger guarantees when used in SSVM training.

the 14-label yeast dataset (Elisseeff & Weston, 2001), which is the largest label space fit using a CRF in Finley & Joachims (2008) and Meshi et al. (2010). Finley & Joachims (2008) analyze the effects of inexact prediction on SSVM training and on test-time prediction. Table 3 considers greedy prediction, loopy belief propagation, exact prediction using an ILP solver, solving the LP relaxation, and SPENs, where the same technique is used at train and test time. All results, besides SPENs, are from Finley & Joachims (2008), which also considers cases where different methods are used in train vs. test. We report hamming error, using 10-fold cross validation.

A key argument of Finley & Joachims (2008) is that SSVM training is more effective when the train-time inference method will not under-generate margin violations. Here, LBP and SPEN, which both approximately minimize a non-convex inference objective, have such a vulnerability, whereas LP does not, since solving the LP relaxation provides a lower bound on the true solution to the value of (10). Since SPEN performs similarly to EXACT and LP, this suggests that perhaps the effect of inexact prediction is more benign for SPENs than for LBP. However, SPENs exhibit alternative expressive power to pairwise CRFs, and thus it is difficult to isolate the effect of SSVM training on accuracy. In future work, we will perform additional experiments to test this.

## 8 Conclusion and Future Work

Structured prediction energy networks employ deep architectures to perform representation learning for structured objects, jointly over both $x$ and $y$. This provides straightforward prediction using gradient descent and an expressive, interpretable framework for the energy function.

We hypothesize that more accurate models can be trained from limited data using the energy-based approach, due to superior parsimony and better opportunities for practitioners to inject domain knowledge. Deep networks have transformed our ability to learn hierarchies of features for the inputs in signal processing problems, such as computer vision and speech recognition. SPENs provide a step in the direction of applying this feature learning revolution to the outputs of structured prediction. Such modeling has the opportunity to improve accuracy in a variety of problems with rich dependencies between outputs.

We have found that SPEN predictions are often spiked at either 0 or 1, despite optimizing a non-convex energy over the set $[0, 1]$. We expect that this results from the energy function being fit to data that is always 0 or 1. We will further study the interplay between the choice of architecture and the integrality of predictions, particularly the piecewise-linear nature of ReLU energy networks.

SPEN prediction requires traversing a complex energy surface using gradient descent. In future work, we will explore alternatives to SSVM training that explicitly model our iterative prediction approach. For example, we can apply the technique of Maclaurin et al. (2015) to update the model parameters by differentiating, with respect to the model parameters $\Psi$, the process of performing iterative gradient-based optimization with respect to $\bar{y}$.

### ACKNOWLEDGMENTS

We thank Nvidia for a generous hardware grant. This work was supported in part by the Center for Intelligent Information Retrieval and in part by NSF grant #CNS-0958392. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

## REFERENCES

Agrawal, Rahul, Gupta, Archit, Prabhu, Yashoteja, and Varma, Manik. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *Proceedings of the 22nd international conference on World Wide Web*, pp. 13–24. International World Wide Web Conferences Steering Committee, 2013.

Beck, Amir and Teboulle, Marc. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

Bhatia, Kush, Jain, Himanshu, Kar, Purushottam, Jain, Prateek, and Varma, Manik. Locally non-linear embeddings for extreme multi-label learning. *CoRR*, abs/1507.02743, 2015. URL `http://arxiv.org/abs/1507.02743`.

Boykov, Yuri and Kolmogorov, Vladimir. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(9):1124–1137, 2004.

Bromley, Jane, Bentz, James W, Bottou, Léon, Guyon, Isabelle, LeCun, Yann, Moore, Cliff, Säckinger, Eduard, and Shah, Roopak. Signature verification using a siamese time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(04):669–688, 1993.

Bucak, Serhat S, Mallapragada, Pavan Kumar, Jin, Rong, and Jain, Anil K. Efficient multi-label ranking for multi-class learning: application to object recognition. In *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 2098–2105. IEEE, 2009.

Bucak, Serhat Selcuk, Jin, Rong, and Jain, Anil K. Multi-label learning with incomplete class assignments. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 2801–2808. IEEE, 2011.

Cabral, Ricardo S, Torre, Fernando, Costeira, João P, and Bernardino, Alexandre. Matrix completion for multi-label image classification. In *Advances in Neural Information Processing Systems*, pp. 190–198, 2011.

Carreira, Joao, Agrawal, Pulkit, Fragkiadaki, Katerina, and Malik, Jitendra. Human pose estimation with iterative error feedback. *arXiv preprint arXiv:1507.06550*, 2015.

Collobert, Ronan, Weston, Jason, Bottou, Léon, Karlen, Michael, Kavukcuoglu, Koray, and Kuksa, Pavel. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

Domke, Jens. Learning graphical model parameters with approximate marginal inference. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(10):2454–2467, 2013.

Elisseeff, André and Weston, Jason. A kernel method for multi-labelled classification. In *Advances in neural information processing systems*, pp. 681–687, 2001.

Finley, Thomas and Joachims, Thorsten. Training structural svms when exact inference is intractable. In *Proceedings of the 25th international conference on Machine learning*, pp. 304–311. ACM, 2008.

Gatys, Leon A., Ecker, Alexander S., and Bethge, Matthias. A neural algorithm of artistic style. *CoRR*, abs/1508.06576, 2015a. URL `http://arxiv.org/abs/1508.06576`.

Gatys, Leon A., Ecker, Alexander S., and Bethge, Matthias. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems 28 (NIPS)*. 2015b.

Ghamrawi, Nadia and McCallum, Andrew. Collective multi-label classification. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pp. 195–200. ACM, 2005.

Globerson, Amir and Jaakkola, Tommi S. Fixing max-product: Convergent message passing algorithms for map lp-relaxations. In *Advances in neural information processing systems*, pp. 553–560, 2008.

Godbole, Shantanu and Sarawagi, Sunita. Discriminative methods for multi-labeled classification. In *Advances in Knowledge Discovery and Data Mining*, pp. 22–30. Springer, 2004.

Goodfellow, Ian J, Shlens, Jonathon, and Szegedy, Christian. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Hariharan, Bharath, Zelnik-Manor, Lihi, Varma, Manik, and Vishwanathan, Svn. Large scale max-margin multi-label classification with priors. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 423–430, 2010.

Hershey, John R, Roux, Jonathan Le, and Weninger, Felix. Deep unfolding: Model-based inspiration of novel deep architectures. *arXiv preprint arXiv:1409.2574*, 2014.

Hsu, Daniel, Kakade, Sham, Langford, John, and Zhang, Tong. Multi-label prediction via compressed sensing. In *NIPS*, volume 22, pp. 772–780, 2009.

Huang, Zhiheng, Xu, Wei, and Yu, Kai. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991, 2015. URL `http://arxiv.org/abs/1508.01991`.

Jasinska, Kalina and Dembczyski, Krzysztof. Consistent label tree classifiers for extreme multi-label classification. In *ICML 2015 Workshop on Extreme Classification*, 2015.

Jernite, Yacine, Rush, Alexander M., and Sontag, David. A fast variational approach for learning markov random field language models. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pp. 2209–2217, 2015.

Ji, Shuiwang and Ye, Jieping. Linear dimensionality reduction for multi-label classification. In *IJCAI*, volume 9, pp. 1077–1082, 2009.

Jordan, Michael I, Ghahramani, Zoubin, Jaakkola, Tommi S, and Saul, Lawrence K. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

Kapoor, Ashish, Viswanathan, Raajay, and Jain, Prateek. Multilabel classification using bayesian compressed sensing. In *Advances in Neural Information Processing Systems*, pp. 2645–2653, 2012.

Koller, Daphne and Friedman, Nir. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Kulesza, Alex and Pereira, Fernando. Structured learning with approximate inference. In *Advances in neural information processing systems*, pp. 785–792, 2007.

Lafferty, John D, McCallum, Andrew, and Pereira, Fernando CN. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pp. 282–289, 2001.

Le, Quoc and Mikolov, Tomas. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pp. 1188–1196, 2014.

LeCun, Yann, Chopra, Sumit, Hadsell, Raia, Ranzato, M, and Huang, F. A tutorial on energy-based learning. *Predicting structured data*, 1:0, 2006.

Lin, Victoria (Xi), Singh, Sameer, He, Luheng, Taskar, Ben, and Zettlemoyer, Luke. Multi-label learning with posterior regularization. In *NIPS Workshop on Modern Machine Learning and Natural Language Processing*, 2014.

Long, Jonathan, Shelhamer, Evan, and Darrell, Trevor. Fully convolutional networks for semantic segmentation. *CVPR (to appear)*, November 2015.

Maclaurin, Dougal, Duvenaud, David, and Adams, Ryan P. Gradient-based hyperparameter optimization through reversible learning. In *Proceedings of the 32nd International Conference on Machine Learning*, July 2015.

Meshi, Ofer, Sontag, David, Globerson, Amir, and Jaakkola, Tommi S. Learning efficiently with approximate inference via dual losses. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 783–790, 2010.

Mordvintsev, Alexander, Olah, Christopher, and Tyka, Mike. Inceptionism: Going deeper into neural networks, June 2015. URL `http://googleresearch.blogspot.com/2015/06/inceptionism-going-deeper-into-neural.html`.

Niculescu-Mizil, Alexandru and Abbasnejad, Ehsan. Label filters for large scale multilabel classification. In *ICML 2015 Workshop on Extreme Classification*, 2015.

Petterson, James and Caetano, Tibério S. Submodular multi-label learning. In *Advances in Neural Information Processing Systems*, pp. 1512–1520, 2011.

Read, Jesse, Pfahringer, Bernhard, Holmes, Geoff, and Frank, Eibe. Classifier chains for multi-label classification. *Machine learning*, 85(3):333–359, 2011.

Srikumar, Vivek and Manning, Christopher D. Learning distributed representations for structured output prediction. In *Advances in Neural Information Processing Systems*, pp. 3266–3274, 2014.

Sutton, Charles and McCallum, Andrew. An introduction to conditional random fields. *Machine Learning*, 4(4):267–373, 2011.

Szegedy, Christian, Zaremba, Wojciech, Sutskever, Ilya, Bruna, Joan, Erhan, Dumitru, Goodfellow, Ian, and Fergus, Rob. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. URL `http://arxiv.org/abs/1312.6199`.

Taskar, B., Guestrin, C., and Koller, D. Max-margin Markov networks. *NIPS*, 2004.

Tsochantaridis, Ioannis, Hofmann, Thomas, Joachims, Thorsten, and Altun, Yasemin. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 104. ACM, 2004.

Tsoumakas, Grigorios and Katakis, Ioannis. Multi-label classification: An overview. *Dept. of Informatics, Aristotle University of Thessaloniki, Greece*, 2006.

Vinyals, Oriol, Kaiser, Lukasz, Koo, Terry, Petrov, Slav, Sutskever, Ilya, and Hinton, Geoffrey. Grammar as a foreign language. In *CoRR.*, 2014.

Xu, Linli, Wang, Zhen, Shen, Zefan, Wang, Yubo, and Chen, Enhong. Learning low-rank label correlations for multi-label classification with missing labels. In *Data Mining (ICDM), 2014 IEEE International Conference on*, pp. 1067–1072. IEEE, 2014.

Yu, Hsiang-Fu, Jain, Prateek, Kar, Purushottam, and Dhillon, Inderjit S. Large-scale multi-label learning with missing labels. In *International Conference on Machine Learning (ICML)*, volume 32, jun 2014.

Zhang, Min-Ling and Zhou, Zhi-Hua. Multilabel neural networks with applications to functional genomics and text categorization. *Knowledge and Data Engineering, IEEE Transactions on*, 18 (10):1338–1351, 2006.

Zheng, Shuai, Jayasumana, Sadeep, Romera-Paredes, Bernardino, Vineet, Vibhav, Su, Zhizhong, Du, Dalong, Huang, Chang, and Torr, Philip. Conditional random fields as recurrent neural networks. In *International Conference on Computer Vision (ICCV)*, 2015.

# A Appendix

## A.1 Details for Improving Efficiency and Accuracy of SPENs

Various tricks of the trade from the deep learning literature, such as momentum, can be applied to improve the prediction-time optimization performance of our entropic mirror descent approach described in Section 2, which are particularly important because $E_x(\bar{y})$ is generally non-convex.

We perform inference in minibatches in parallel on GPUs.

When 'soft' predictions are useful, it can be useful to augment $E_x(\bar{y})$ with an extra term for the entropy of $\bar{y}$. This can be handled at essentially no computational cost, by simply normalizing the iterates in entropic mirror descent at a certain 'temperature.' This is only done at test time, not in the inner loop of learning.

Typically, backpropagation computes the gradient of output with respect to the input and also computes the gradient of the output with respect to any parameters of the network. For us, however, we only care about gradients with respect to the inputs $\bar{y}$ during inference. Therefore, we can obtain a considerable speedup by avoiding computation of the parameter gradients.

We train the local energy network first, using a local label-wise prediction loss. Then, we clamp the parameters of the local energy network and train the label energy network. Finally, we perform an additional pass of training, where all parameters are updated using a small learning rate.

## A.2 Hyperparameters

For prediction, both at test time and in the inner loop of learning, we ran gradient descent with momentum = 0.95, a learning rate of 0.1, and no learning rate decay. We terminated prediction when either the relative change in the objective was below a tolerance or the $l_\infty$ change between iterates was below an absolute tolerance.

For training, we used sgd with momentum 0.9 with learning rate and learning rate decay tuned on development data. We use l2 regularization both when pre-training the features and net and during SSVM training, with l2 weights tuned on development data.

We did not tune the sizes of the hidden layers for the feature network and label energy network. These were set based on intuition and the size of the data, the number of training examples, etc.

| | #labels | #features | # train | % true labels |
|---|---|---|---|---|
| Bibtex | 159 | 1836 | 4880 | 2.40 |
| Delicious | 983 | 500 | 12920 | 19.02 |
| Bookmarks | 208 | 2150 | 60000 | 2.03 |
| Yeast | 14 | 103 | 2417 | 30.3 |

Table 4: Properties of the datasets.

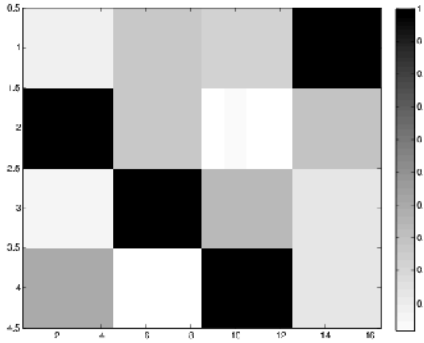


Figure 2: Structure learning on synthetic task using 10% of the data. The measurement matrix still recovers interactions between the labels characteristic of the data generating process