Structured Energy Network as a Loss Function

Jay-Yoon Lee[¶]
lee.jayyoon@snu.ac.kr

Dhruvesh Patel[♣] **Purujit Goyal**[♣] **Wenlong Zhao**[♣]
dpatel, purujitgoyal, wenlongzhao@cs.umass.edu

Zhiyang Xu[◊]
zhiyangx@vt.edu

Andrew McCallum[♣]
mccallum@cs.umass.edu

[¶]Graduate School of Data Science, Seoul National University
[♣]Manning College of Information & Computer Sciences, University of Massachusetts Amherst
[◊]Department of Computer Science, Virginia Tech

Abstract

Belanger & McCallum (2016) and Gygli et al. (2017) have shown that energy networks can capture arbitrary dependencies amongst the output variables in structured prediction; however, their reliance on gradient based inference (GBI) makes the inference slow and unstable. In this work, we propose Structured Energy As Loss (SEAL) to take advantage of the expressivity of energy networks without incurring the high inference cost. This is a novel learning framework that uses an energy network as a trainable loss function (loss-net) to train a separate neural network (task-net), which is then used to perform inference through a forward pass. We establish SEAL as a general framework wherein various learning strategies like margin-based, regression, and noise-contrastive could be employed to learn the parameters of loss-net. Through extensive evaluation on multi-label classification, semantic role labeling, and image segmentation, we demonstrate that SEAL provides various useful design choices, is faster at inference than GBI, and leads to significant performance gains over the baselines.

1 Introduction

Structured prediction is an important machine learning problem in which, given an input $\mathbf{x}$, the model needs to predict the output $\mathbf{y} \in \mathcal{Y}_1 \times \mathcal{Y}_2 \times \cdots \times \mathcal{Y}_L$ where there are dependencies across $y_1, \ldots, y_L$. Common structured prediction tasks include multi-label classification (Belanger & McCallum, 2016; Gygli et al., 2017), extracting parse trees or semantic role labels from texts (Palmer et al., 2010), and image segmentation (Müller, 2014). Traditionally, these tasks are performed using models that explicitly represent the dependencies among output variables, such as directed and undirected graphical models, including Hidden Markov Models and Conditional Random Fields. However, except in cases with very limited dependency patterns (Lafferty et al., 2001; Ghamrawi & McCallum, 2005), inference algorithms in these methods are difficult to engineer and expensive to run because structured output spaces are exponentially large. In a multi-label classification task, for example, the size of the label space $\mathcal{Y}$ is 2^L given $\mathcal{Y}_i = \{0, 1\}$.

With their growing expressivity and capacity, it is common to use large feedforward neural networks to implicitly capture dependencies among output variables. Given an input $\mathbf{x}$, the elements of an output $\mathbf{y}$ are predicted in a conditionally independent manner in parallel, solely relying on the representation of input $\mathbf{x}$ to capture dependencies among output variables. While the feedforward approach is computationally efficient, the widely used cross-entropy loss does not directly capture

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

dependencies in the output space and thus often lacks statistical efficiency in capturing multivariate correlations.

A recent line of methods (Belanger & McCallum, 2016; Gygli et al., 2017; Rooshenas et al., 2019) model the joint space of $\mathbf{x}$ and multivariate $\mathbf{y}$ explicitly with an energy function $E_\Theta(\mathbf{x}, \mathbf{y})$ (LeCun et al., 2006), which aims to learn arbitrary global dependencies in the output space. They relax the combinatorial output space $\mathcal{Y}$ to $\tilde{\mathcal{Y}} = [0,1]^L$ and apply gradient-based inference (GBI) directly on $\tilde{\mathbf{y}}$ to search for optima on the relaxed energy surface as $\arg\min_{\tilde{\mathbf{y}}} E(\mathbf{x}, \tilde{\mathbf{y}})$. These energy models are therefore often called structured *prediction* energy networks (SPENs).

SPENs show noticeable gains in predictive performance over feedforward models and traditional graphical models such as (Chen et al., 2015; Schwing & Urtasun, 2015). Due to the use of GBI, however, inference for SPENs remains relatively inefficient when compared to feedforward approaches (Tu & Gimpel, 2019). Moreover, models using GBI are often difficult to train and sensitive to sampling methods and numerous hyperparameters, including step size, number of iterations, and initialization for GBI. This raises a question: *Can energy networks be used in a way that is as expressive as SPENs, as efficient at inference as feedforward approaches, and also easy to train?* To this end, we propose using structured energy networks as parameterized loss functions for feedforward networks, instead of as prediction networks.

In this paper, we introduce the **S**tructured **E**nergy **A**s **L**oss (SEAL) framework which uses trainable SPENs (loss-net[1]) as loss functions to guide the training of other feedforward networks (task-net). The key idea is to provide the feedforward network with access to rich dependencies in the output space through a learned loss-net (§2.1 SEAL-static). We further propose to learn the loss-net in a dynamic fashion (§2.2 SEAL-dynamic) by adjusting the loss-net to be tailored to the most up-to-date outputs of the feedforward network; and then propose an NCE ranking loss uniquely suited for the SEAL-dynamic framework. Using feature-based multi-label classification as a case study, we extensively analyze the impact of various loss functions for updating the loss-net (§2.3). We further demonstrate the faster inference and higher performance of SEAL, compared to other baseline methods, over feature-, text-based multi-label classification, semantic role labeling, and image segmentation (§4). Additionally, we show that energy models capture the dependencies on the output space through analyzing gradient signals in an ablation study (Appendix G).

2 Structured Energy Network as Loss (SEAL)

Notation. $\mathcal{X}$ denotes the input space, $\mathcal{Y} = \{0,1\}^L$ the label space, and $\tilde{\mathcal{Y}} = [0,1]^L$ the continuous relaxation of $\mathcal{Y}$. Let $\mathcal{D} = \{(\mathbf{x}^{(j)}, \mathbf{y}^{(j)}) \in \mathcal{X} \times \mathcal{Y}\}_{j=1}^N$ denote the dataset, and $y_i \in \{0,1\}$ the i-th dimension in the label vector $\mathbf{y}$ for $i \in \{1, \ldots, L\}$, and $\tilde{y_i} \in [0,1]$ its continuous relaxation. The feedforward network $F_\Phi : \mathcal{X} \to \tilde{\mathcal{Y}}$ is a parametrized function that maps an input $\mathbf{x} \in \mathcal{X}$ to an output $F_\Phi(\mathbf{x}) \in \tilde{\mathcal{Y}}$, where $F_\Phi(\mathbf{x})_i$ is the predicted probability of label i occurring, given the input $\mathbf{x}$. The structured energy network $E_\Theta : \mathcal{X} \times \tilde{\mathcal{Y}} \to \mathbb{R}$ is a parameterized function that maps an input $\mathbf{x}$ and a continuously relaxed output $\tilde{\mathbf{y}}$ to real-valued scalar energy. For clarity, we refer to the feedforward network as *task-net*, and the structured energy network as *loss-net*.

As depicted in Figure 1, the proposed SEAL framework consists of two parametric components. First, the loss-net E_Θ is trained by an energy loss L_E to capture an energy surface that reflects the latent dependencies among output variables. Low energy indicates a high quality of a label prediction given an input. Second, a task-net F_Φ is trained using a loss L_F that involves minimizing the energy E_Θ estimated by the loss-net. After training finishes, the loss-net is no longer utilized; inference time predictions are obtained solely from the task-net as $\hat{y}_i = \mathbf{1}(F_\Phi(x)_i \geq 0.5)$. Unlike SPEN which requires iterative gradient-based inference, a task-net trained with the SEAL framework has the same computational efficiency at inference-time as standard feedforward networks, while enjoying a loss that captures the label dependencies through the loss-net (Appendix D).

[1]We refer an energy network as loss-net when it is used as a trainable loss function for another network.

To guide the task-net training with the loss-net E_Θ, we propose the task-net training loss $\mathrm{L}_F(\Phi;\Theta)$ to be a weighted sum of estimated energy and a binary-cross entropy (BCE) loss as

$$\mathrm{L}_F(\Phi;\Theta) = \lambda_1 E_\Theta(\mathbf{x}, F_\Phi(\mathbf{x})) + \lambda_2 \sum_{j=1}^{L} \mathrm{BCE}(y_j, F_\Phi(\mathbf{x})_j) \tag{1}$$

where $(\mathbf{x},\mathbf{y})$ is a training instance and $\mathrm{BCE}(y_j, F_\Phi(\mathbf{x})_j) = -[y_j \log F_\Phi(\mathbf{x})_j + (1 - y_j)\log(1 - F_\Phi(\mathbf{x})_j)]$. By incorporating the energy term in L_F, we enable the loss function to reflect the latent variable dependencies in the output space, whereas BCE treats L labels in a conditionally independent fashion.

We present the formulation of E_Θ used for multi-label classification to analyze the effect of introducing E_Θ into L_F without loss of generality. We use energy network $E_\Theta(x, y)$ that has the same structure with Belanger & McCallum (2016) and Gygli et al. (2017) for a fair comparison with them. The energy E_Θ is the sum of local energy E_Θ^{local} and global energy E_Θ^{global} which are defined as:

$$E_\Theta^{\text{local}}(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{L} y_i \mathbf{b}_i^\top T_E(\mathbf{x}),\ E_\Theta^{\text{global}}(\mathbf{x},\mathbf{y}) = \mathbf{v}^\top \gamma(\mathbf{M}\mathbf{y}), \tag{2}$$

where $\mathbf{b}_i, \mathbf{v}, \mathbf{M}, T_E$ contain learnable parameters, T_E is a feature network to represent an input $\mathbf{x}$ in the loss-net, and $\gamma(z) = \log(1 + e^z)$ is the softplus activation function. We observe from the derivative of the global energy

$$\frac{\partial E_\Theta^{\text{global}}}{\partial y_i} = \left\{M^T \mathrm{Diag}(\gamma'(M\mathbf{y}))\mathbf{v}\right\}_i \tag{3}$$

that all dimensions of $\mathbf{y}$ decides the gradient signal for each label i. The global energy term E_Θ^{global} in L_F, specifically, allows the gradient of the task-net loss to capture the dependency of y_i on all L dimensions of $\mathbf{y}$. In contrast, the gradient of BCE for each label i only depends on the i-th dimension of $\mathbf{y}$. More details are in Appendix A.

2.1 SEAL-static

Since the task-net loss L_F depends on the quality of the loss-net, it is critical to find the loss-net parameters Θ that produce the best loss surface for training the task-net Φ. In order to achieve this, we first estimate the loss-net parameters $\widehat{\Theta}$ over training data and then optimize the task-net parameters Φ by plugging in the fixed loss-net parameters $\widehat{\Theta}$ into L_F. Formally, given training instances $(\mathbf{x},\mathbf{y}) \in \mathcal{D}$, the task-net parameters Φ are trained in the following manner:

$$\min_\Phi \frac{1}{|\mathcal{D}|}\sum_{\mathcal{D}} \mathrm{L}_F(\Phi;\widehat{\Theta}) \qquad \text{s.t. } \widehat{\Theta} = \arg\min_\Theta \frac{1}{|\mathcal{D}|}\sum_{\mathcal{D}} \mathrm{L}_E(\mathbf{x},\mathbf{y},\tilde{\mathbf{y}};\Theta) \tag{4}$$

Since the loss function L_F is static after pretraining $\widehat{\Theta}$, we call this approach SEAL-static. We discuss various types of L_E losses in §2.3.

2.2 SEAL-dynamic

Ideally, we would like to learn a perfect energy surface by estimating $\widehat{\Theta}$ that is static. However, estimating an accurate energy surface over a high-dimensional, continuous joint space $\mathcal{X} \times \tilde{\mathcal{Y}}$ is a challenging task. Instead, we hypothesize that using loss-net to dynamically estimate the energy surface that is focused around the current input and task-net output pair $(\mathbf{x}, F_\Phi(x))$ could be more beneficial in teaching task-net. To this end, we propose SEAL-dynamic which dynamically adapts loss-net so that the region of interest for training task-net is well represented.

Given a batch of training data B_t, a complete training step of SEAL-dynamic is provided as

$$\Theta_t \leftarrow \Theta_{t-1} - \nabla_\Theta \frac{1}{|B_t|}\sum_{B_t} \mathrm{L}_E\left(\mathbf{x},\mathbf{y},F_{\Phi_{t-1}}(\mathbf{x});\Theta\right) \tag{5}$$

$$\Phi_t \leftarrow \Phi_{t-1} - \nabla_\Phi \frac{1}{|B_t|}\sum_{B_t} \mathrm{L}_F(\Phi;\Theta_t) \tag{6}$$

where the loss-net parameters Θ_t and the task-net parameters Φ_t are optimized in alternating fashion. Note that, from Eq.(4) to Eq.(5), arbitrary label samples $\tilde{\mathbf{y}}$ were replaced with $F_{\Phi_{t-1}}(\mathbf{x})$ from the task-net during the training of Θ_t parameters. Thus the energy surface learned by the loss-net is adaptive to the task-net. On the contrary, in SEAL-static, the training of the static loss-net is independent of the task-net. Further, the update step for the task-net parameters Φ_t relies on dynamic loss-net parameters Θ_t rather than static $\widehat{\Theta}$. In §4, we show the effectiveness of the proposed SEAL framework compared to standard feedforward networks and SPENs, and further that SEAL-dynamic often improves over SEAL-static.

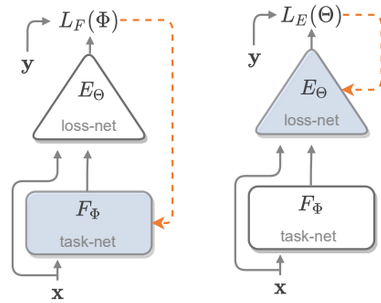


Figure 1: Overview of the SEAL framework. The figure on the left describes the update to the task-net in Eq.(4) and Eq.(6) and the figure on the right describes the update to the loss-net in Eq.(5).

Algorithm 1: SEAL-dynamic Algorithm

Require: $(\mathbf{x}, \mathbf{y})$: Training Instance
Require: F_Φ: Feedforward Network
Require: `sampling`: True/False flag
Require: optimizer$_\Theta$, optimizer$_\Phi$
Require: T: No. of steps
  $\Theta_0, \Phi_0 \leftarrow$ Random initialization, $t \leftarrow 0$
  while $t < T$ **do**
    if `sampling` **then**
      $S \leftarrow \{\mathbf{y}^{(i)}, i = 1, \ldots, K | \mathbf{y}^{(i)} \sim F_{\Phi_t}(\mathbf{x})\}$
    else
      $S \leftarrow \{F_{\Phi_t}(\mathbf{x})\}$ ▷ singleton set
    $t \leftarrow t + 1$
    Update Θ as Eqn. 5 using $\tilde{\mathbf{y}} \in S$ and optimizer$_\Theta$
    Update Φ as Eqn. 6 using optimizer$_\Phi$

2.3 Energy losses L_E

We now discuss various energy loss functions L_E with different characteristics such as margin-based, regression-based, and contrastive-sampling-based, that can be plugged into the SEAL framework to train the loss-net. For brevity, we use $\mathrm{L}_E(\Theta)$ and $\mathrm{L}_E(\mathbf{x}, \mathbf{y}, \tilde{\mathbf{y}}; \Theta)$ interchangeably.

Margin-based ($\mathbf{L}_{E-\text{margin}}$). SPEN (Belanger & McCallum, 2016) learns structured energy network with the SSVM loss (Taskar et al., 2004; Tsochantaridis et al., 2004) so that energy function learns to predict sufficient energy difference, larger than the margin $\Delta(\tilde{\mathbf{y}}, \mathbf{y})$, between an arbitrary output $\tilde{\mathbf{y}}$ and a true output $\mathbf{y}$. To examine the effect of the margin-based loss in SEAL, we follow SPEN and utilize the SSVM loss as

$$\mathrm{L}_{E-\text{margin}} = \sum_{\mathbf{x},\mathbf{y}} \max_{\tilde{\mathbf{y}}} \left[\Delta(\tilde{\mathbf{y}}, \mathbf{y}) - E_\Theta(\mathbf{x}, \tilde{\mathbf{y}}) + E_\Theta(\mathbf{x}, \mathbf{y})\right]_+ . \tag{7}$$

SPEN resorts to gradient ascent on $\tilde{\mathbf{y}}$ to solve the optimization problem $\arg\max_{\tilde{\mathbf{y}}} \Delta(\tilde{\mathbf{y}}, \mathbf{y}) - E_\Theta(\mathbf{x}, \tilde{\mathbf{y}})$. InfNet (Tu & Gimpel, 2018) proposes an adversarial training of the task-net Φ together with the energy network Θ. We identify InfNet as a special case of SEAL-dynamic that uses $\mathrm{L}_{E-\text{margin}}$. We defer a detailed description of it to the related work (§3).

Regression-based ($\mathbf{L}_{E-\text{regression}}$). Deep Value Network (DVN) (Gygli et al., 2017) attempts to learn an energy network which directly outputs a score that is similar to the metric $s(\tilde{\mathbf{y}}, \mathbf{y})$ of interest, such as F1 score, that compares arbitrary $\tilde{\mathbf{y}}$ with true output $\mathbf{y}$. Following DVN, we let the true score $s(\cdot)$[2] and $\sigma(-E(\cdot))$ be between $[0, 1]$ and express the regression-based loss as cross entropy between s and $\sigma(-E)$:

$$\mathrm{L}_{E-\text{regression}} = -s(\tilde{\mathbf{y}}, \mathbf{y}) \log \sigma\left(-E_\Theta(\mathbf{x}, \tilde{\mathbf{y}})\right) - \left(1 - s(\tilde{\mathbf{y}}, \mathbf{y})\right) \log\left(1 + \sigma\left(-E_\Theta(\mathbf{x}, \tilde{\mathbf{y}})\right)\right)$$

where $\sigma(\cdot)$ is a sigmoid function. DVN utilizes two samples to train the model: the model prediction that minimizes the energy and an adversarial sample $\tilde{\mathbf{y}}$ that maximizes the loss $\mathrm{L}_{E-\text{regression}}$. We follow

[2] In this paper, we adopt soft F1 score $s(\tilde{\mathbf{y}}, \mathbf{y})$ from Gygli et al. (2017) that is defined on the continuous $\tilde{\mathbf{y}} \in [0, 1]^L$.

this approach for training SEAL-static with $\mathrm{L}_{E-\text{regression}}$. To train SEAL-dynamic with $\mathrm{L}_{E-\text{regression}}$, we simply use the output of task-net, $F_{\Phi_{t-1}}(\mathbf{x})$, as described in Eq.(5).

To better capture the energy surface in SEAL-dynamic, we define the *regression-s* loss $\mathbf{L}_{E-\textbf{regression-s}} = \sum_{\tilde{\mathbf{y}} \in S} \mathrm{L}_{E-\text{regression}}(\mathbf{x}, \mathbf{y}^*, \tilde{\mathbf{y}}; \Theta)$ where the sample set S is taken by continuously perturbing task-net output $F_{\Phi_{t-1}}(\mathbf{x})$ with Gaussian noise (experimental analysis on the 'effect of samples' is in §5).

Noise-contrastive ranking ($\mathbf{L}_{E-\text{NCE}}$). Lastly, we propose to use noise contrastive estimation (NCE) ranking loss (Ma & Collins, 2018) uniquely for SEAL-dynamic. We are interested in capturing appropriate energy surface in high-probability output spaces of task-net. To reflect such space accurately, we propose to train the loss-net E_Θ to contrast the true output $\mathbf{y}$ with a group of K negative samples drawn from the task-net prediction $F_\Phi(\mathbf{x})$ motivated by NCE.

Before we discuss $\mathrm{L}_{E-\text{NCE}}$ for SEAL-dynamic in more detail, we first review the original form of NCE ranking loss from Ma & Collins (2018). For K samples $\mathbf{y}^{(k)} \sim P_N,\ k = 1, \ldots, K$, drawn from a noise distribution P_N, rewriting ground truth $\mathbf{y}$ as $\mathbf{y}^{(0)}$ without loss of generality, the NCE ranking loss is defined as

$$\mathrm{L}_{E-\text{NCE}} = -\log \frac{\exp\left(s(\mathbf{x}, \mathbf{y}^{(0)}; \Theta)\right)}{\sum_{k=1}^{K} \exp\left(s(\mathbf{x}, \mathbf{y}^{(k)}; \Theta)\right)} \qquad \text{where } \ s(\mathbf{x}, \mathbf{y}; \Theta) = -E_\Theta(\mathbf{x}, \mathbf{y}) - \log P_{\mathrm{N}}(\mathbf{y}).$$

Given the noise samples $y^{(k)}$, minimizing Eq.(8) minimizes the energy of the ground truth sample $\mathbf{y}^{(0)}$ while increasing the energy of the noise samples.

In SEAL-dynamic, we propose to use $\prod_i P(y_i|\mathbf{x}; \Phi_t)$ which is dynamically changing in place of the static $P_N(\mathbf{y})$ in Eq.(8). The NCE ranking loss is thus

$$\text{Eq.(8) where } \ s(\mathbf{x}, \mathbf{y}; \Theta) = -E_\Theta(\mathbf{x}, \mathbf{y}) - \sum_i \log P(y_i|\mathbf{x}; \Phi_t). \tag{8}$$

The novelty of this proposal is that we view the output of a task-net as a dynamic noise distribution, and that the loss-net in turn teaches the task-net that generates the contrastive noise samples (Eq.(5)).

The NCE ranking loss $\mathrm{L}_{E-\text{NCE}}$ brings two benefits in SEAL-dynamic. First, $\mathrm{L}_{E-\text{NCE}}$ captures the output distribution of the task-net rather than just a single point. Second, $\mathrm{L}_{E-\text{NCE}}$ allows estimation of a more fine-grained loss surface as the training steps proceed. In our SEAL-dynamic learning procedure, we hypothesize that $\prod_i P(y_i|\mathbf{x}; \Phi_t)$ becomes closer to the true data distribution as training steps proceed. Contrasting these noise samples to the ground truth could lead the loss surface estimated by loss-net to be more fine-grained. This is in line with the known fact that NCE methods work the best when the noise distribution is close to the data distribution but not the same (Gutmann & Hirayama, 2012).

To thoroughly experiment with the proposed NCE ranking loss $\mathrm{L}_{E-\text{NCE}}$, we also train a static energy network with $\mathrm{L}_{E-\text{NCE}}$ in order to utilize it as a prediction network with GBI and as a loss-net in SEAL-static. We set $P_N(\mathbf{y}) = \prod_i P(y_i|\mathbf{x}; \Psi)$ in Eq.(8), where Ψ is a fixed pretrained feedforward network. When we train an energy network E_Θ with $\mathrm{L}_{E-\text{NCE}}$, $\frac{\exp(-E_\Theta(\mathbf{x},\mathbf{y}))}{\sum_{\mathbf{y} \in \mathcal{Y}} \exp(-E_\Theta(\mathbf{x},\mathbf{y}))}$ becomes an unbiased estimator of the true distribution $P(\mathbf{y}|\mathbf{x})$ (Ma & Collins, 2018). Given a perfect energy $E_\Theta(\mathbf{x}, \mathbf{y})$, minimizing it with respect to $\mathbf{y}$ in SEAL-static and for prediction with GBI is equivalent to minimizing the estimate of $-\log P(\mathbf{y}|\mathbf{x})$.

3 Related Work

Learning with dynamic loss. Wu et al. (2018) and Huang et al. (2019) attempt to learn a dynamic loss for multi-class classification tasks. Both approaches try to learn a parametrized loss function that can directly increase the model's metric score m, such as the BLEU or 0-1 accuracy, on the validation set. Since m is usually non-differentiable, Huang et al. (2019) view this problem as a reinforcement learning problem and attempt to learn a loss function that maximizes the expected reward. On the other hand, Wu et al. (2018) relax the m to be differentiable and try to learn a loss parameter (Θ) that can improve the relaxed score on feed-forward (Φ) output.

There are three major differences between SEAL and the previous work. While both Wu et al. (2018); Huang et al. (2019) use the parametric form $\sigma(\mathbf{y}^T \Theta \log F_\Phi(x))$ to define the loss function, SEAL

utilizes a bigger class of functions expressed using general energy network E_Θ. Moreover, SEAL does not require manually designed state vectors and works directly with the task-net outputs. Lastly, while the SEAL framework is designed for structured prediction, Wu et al. (2018) and Huang et al. (2019) are designed for multi-class classification and it is non-trivial to extend these methods for structured prediction.

Structured Prediction Energy Network (SPEN). Structured prediction energy networks (Belanger & McCallum, 2016) and its variants (Gygli et al., 2017; Rooshenas et al., 2019) (together referred to as SPENs) learn an energy network $E_\Theta : \mathcal{X} \times \tilde{\mathcal{Y}} \rightarrow \mathbb{R}$ and predict the output using gradient-based inference (GBI) with the objective $\mathbf{y} = \arg\min_y E(\mathbf{x}, \mathbf{y})$. The optimization objective for training the SPENs relies on finding adversarial or margin-violating points in the label space using gradient based inference (GBI). This learning process in SPENs is highly sensitive with respect to hyper-parameters such as the learning rate and initial points for the GBI, resulting in high variance in the model performance. Nonetheless, SPENs are strong baselines for structured prediction tasks and are used as motivations for the design of SEAL. They are further discussed in §2.3.

Jointly learning Θ and Φ. Inference Network (InfNet) (Tu & Gimpel, 2018) is the first work that interactively learns a structured energy network and a feedforward network. It aims to replace the expensive GBI in SPEN with a feedforward network's forward pass. During its training with the SSVM loss (Eq.(7)), SPEN maximizes $\Delta(\tilde{\mathbf{y}}, \mathbf{y}) - E_\Theta(\mathbf{x}, \tilde{\mathbf{y}})$ with respect to $\tilde{\mathbf{y}}$ by GBI. InfNet, on the contrary, trains a feedforward network F_Φ to generate $\tilde{\mathbf{y}} = F_\Phi(x)$ that maximizes the term. This results in an adversarial framework (Goodfellow et al., 2014), where a feedforward network maximizes the margin-based loss while an energy network minimizes it. The approach, however, suffers from train-test-time objective mismatch (Tu et al., 2020). While the feedforward network serves as an adversarial sampler during training, the end goal is to learn a feedforward network that predicts outputs with low energy at test time. Tu et al. (2020) resolves this by learning two separate feed-forward networks: one adversarial sampler for training and the other for performing inference.

Our proposed SEAL generalizes, and differs from, InfNet by providing a framework that can employ completely independent loss functions for training loss-net and task-net. SEAL updates task-net in the direction of reducing energy defined by loss-net; loss-net is only used as a tool for evaluation of loss, and how loss-net is trained does not matter as long as it estimates a good loss surface for task-net. The SEAL framework is not interested in producing the similar test-time behaviour of SPEN as InfNet did. Despite the difference in the end goal, mechanically speaking, InfNet is a special case of SEAL-dynamic (i.e. SEAL-dynamic with $\mathrm{L}_{E-\text{margin}}$) if the margin is removed from the SSVM loss for the feedforward model training. In fact, Tu et al. (2020) reports that this special case leads to more efficient and stable learning of the feedforward network in their ablation study. We use this special case of InfNet to represent SEAL with $\mathrm{L}_{E-\text{margin}}$ in the main experiment. By doing so, we reinterpret the success of InfNet as an improvement from utilizing energy network as a loss function, as our formulations show, rather than task-net mimicking the test-time behaviour of energy network.

4 Experiments

Tasks and Datasets. We demonstrate the effectiveness of SEAL on widely deployed structured prediction tasks: feature and text-based multi-label classification (MLC), semantic role labeling (SRL), and image segmentation. Feature-based MLC is a standard evaluation task in previous works on SPENs (Belanger & McCallum, 2016; Gygli et al., 2017; Tu & Gimpel, 2018). We utilize 7 feature-based MLC datasets which cover various label sizes (ranging from 27 to 4k), training sizes, and input characteristics (binary, real) as shown in Table 5 to extensively compare baselines and proposed SEAL approaches: cross-entropy (CE), energy network, SEAL-static, and SEAL-dynamic. Then we verify that SEAL-dynamic, which outperforms SEAL-static on feature-based MLC, can also effectively train large pre-trained language models like BERT (Devlin et al., 2019) on text-based MLC and SRL tasks and convolutional networks on image segmentation. Dataset statistics are detailed in Appendix B.

Feature network, loss-net and task-net. In MLC and SRL experiments, to represent input $\mathbf{x}$, we utilize the same feature network structures T_E and T_F for the energy and feedforward networks, which correspond to the loss-net and the task-net for SEAL framework. The energy network utilizes T_E in the manner shown in equation 2. Given a feature network T_F that generates features in $\mathbb{R}^h$,

Table 1: Performance (F1 ↑) of feature-based MLC datasets. The task-net learned with SEAL-static and SEAL-dynamic outperforms cross-entropy-trained model most of the time. The SEAL framework which utilizes an energy network as a loss performs better and faster than an energy network utilized as a prediction network. Best results per dataset are marked with underline; best results within a framework (energy network with GBI, SEAL-static, SEAL-dynamic) are marked in **bold**.

method	samples	discrete input datasets			continuous input datasets				mean F1 gain over CE	
		bibtex	**delicious**	**genbase**	**cal500**	**eurlexev**	**expr_fun**	**spo_fun**	**overall**	**−-genbase**
cross-entropy (CE)	x	42.40	29.89	47.37	33.58	42.19	37.50	27.81	-	-
energy network with GBI										
SPEN	x	42.99	24.20	32.13	37.24	**41.86**	**36.74**	27.95	-2.52	-0.40
DVN	x	**45.95**	**24.87**	**77.92**	**47.74**	25.49	31.47	**29.35**	3.15	-1.42
NCE ranking	o	12.95	12.69	12.40	33.89	0.19	27.74	18.06	-20.40	-17.98
SEAL-static										
margin	x	**43.11**	28.08	57.45	33.91	42.15	**38.13**	28.15	1.46	0.02
regression	x	42.29	**30.09**	**96.68**	37.63	42.18	38.12	**28.42**	7.78	0.86
NCE ranking	o	43.03	30.08	96.43	**37.82**	**42.11**	37.78	28.29	7.83	0.95
SEAL-dynamic										
margin (InfNet)	x	42.86	29.75	96.53	36.69	41.83	37.81	28.43	7.59	0.67
regression	x	43.74	29.79	96.95	37.97	41.67	37.99	**29.02**	8.05	1.13
regression-s	o	44.53	29.87	96.81	38.95	42.37	37.93	28.29	8.29	1.43
NCE ranking	o	**44.76**	**34.67**	**97.32**	**41.62**	**42.77**	**38.28**	28.83	**9.64**	**2.93**
Inference speed (examples/sec)	feedforward	840	2307	1005	1080	317	3801	5231	Average speedup	3.63x
	GBI	414	599	574	438	162	638	709		

the task-net $F_\Phi(x) = \mathbf{G}T_F(x)$, where $\mathbf{G} \in \mathbb{R}^{L \times h}$ is a matrix consisting of learnable embeddings, one row of embedding for each label. Specific feature network structures vary by tasks: multi-layer perceptron for feature-based MLC and BERT for text-based MLC and SRL. In image segmentation experiments, we follow Gygli et al. (2017) and use a Fully Convolutional Network (FCN) as the task-net; the loss-net uses the same convolutional layers in the FCN, followed by fully connected layers which replaces the tranposed convolutional layers in FCN to output scalar scores, instead of the energy formulation in Eq.(2).

Training. We use separate ADAM optimizers (Kingma & Ba, 2014) for the loss-net (Θ) and the task-net (Φ) which optimize parameters in alternating fashion[3]. We defer specific training details such as hyperparameters and gpu environment to the Appendix C.1.

4.1 Multi-label classification for feature-based datasets

Multi-label classification (MLC) is a structured prediction task that does not hold an obvious latent structure in the multivariate space. In addition to including Bibtex and Delicious that were used in previous works (Belanger & McCallum, 2016; Gygli et al., 2017; Tu & Gimpel, 2018), we carefully picked smaller (Genbase, Cal500) and larger datasets (Eurlex-ev) as well as datasets with known taxonomy (Spo_FUN, Expr_FUN) to examine SEAL in various scenarios. The experiment results (test F1) across 7 feature-based dataset are presented in Table 1: feedforward network trained with cross-entropy (row 1), energy networks (row 2-4) described in §3 that is evaluated with GBI, task-nets trained with SEAL-static (row 5-7), and task-nets trained with SEAL-dynamic (row 8-11).

In general, SEAL-static and SEAL-dynamic outperform both cross-entropy (CE) and prediction energy networks with GBI on almost all datasets with average gains of SEAL-dynamic over CE ranging from +0.67 to +2.93 F1 points, excluding results on genbase dataset. While the near-perfect performance of SEAL on genbase is interesting (see Appendix E.2 for details), we exclude this excessive gain for computing the average performance when comparing different models throughout this paper.

Since SEAL uses single forward pass of the task-net for inference, it is as fast as the feedforward model and 3.63x faster (on average) at inference than the corresponding energy network trained using GBI. However, in training time, SEAL utilizes about twice the parameter to feedforward model as it loads both task-net and loss-net and takes about 5x more time to train compared to feedforward model. In summary, SEAL is better performing than both energy network and feedforward model and as fast as feedforward model at inference time by utilizing more computations in training time. The training time, inference time, and parameter size of different methods are reported in Appendix D.

[3]The code we used to train and evaluate our models is available a `https://github.com/leejayyoon/SEAL_StructuredEnergyAsLoss`

Cross entropy vs SEAL. While cross entropy cannot capture the dependencies between the labels, we find that global energy of the loss-net visibly captures some of these dependencies. We plot the gradients of global energy with respect to a pair of labels to analyse pairwise dependencies. These plot can be found in the Appendix G.

Energy network vs. SEAL-static. There is a high variability in the performance of the energy network with GBI. For instance, while SPEN and DVN surpass CE models with large margins for half of the datasets, they underperform on the rest of the datasets with similar margins. Further, even though DVN has the best performance on three datasets, its average performance (-genbase) is lower than CE. Nonetheless, in SEAL-static, when the identical energy network is utilized as a loss-net, the performances increase in a stable manner (0.42-2.26 F1 on average) with faster (1.8x-7.4x) inference time. This trend is most evident with the NCE ranking loss, where the energy network trained with it drastically fails to provide a good energy surface for predicting output with GBI, whereas it provides a useful loss surface for training task-net.

We believe there are two factors for this stable improvement of SEAL-static. One is that injection of an energy surface as a loss is a softer injection of gradients compared to it being used as a surface for inference. The second factor is a synergy between cross-entropy and the energy surface. Even when certain regions of the energy surface are less accurate, the cross-entropy signal could compensate for the erroneous surface. Thus, we claim that an energy network as a loss is a more stable way of utilizing it compared to GBI.

SEAL-static vs. SEAL-dynamic. As seen in Table 1, with an average gain of 0.47 to 1.97 F1 points, SEAL-dynamic is always on par or better than the corresponding SEAL-static. This confirms our hypothesis that it is helpful for the loss surface to consider the current state of the task-net as opposed to estimating the global energy landscape. Lastly, SEAL-dynamic not only brings better performance but also simplifies the engineering as both loss-net and task-net can be trained from scratch. We observed that binary cross entropy in L_F stabilizes the learning signal even in the early stage when loss-net might not be well-trained.

In summary, the SEAL framework outperforms CE loss and energy networks with GBI. We observe the generality of the SEAL framework as all of the energy loss types benefit working with SEAL, although more effective types of energy loss exist. SEAL-dynamic brings extra gain in performance to SEAL-static and among them, the NCE ranking loss shows the most notable improvement. Lastly, as can be seen with the example of the NCE ranking loss, an energy network that serves as a poor prediction network does not necessarily correlate to a poor energy loss for SEAL.

4.2 Multi-label classification for large text datasets

To test the effectiveness of SEAL on large language models, we finetune pre-trained BERT as the feature network T_E and T_F on text MLC datasets. Considering the computational expense of running hyper-parameter searches on BERT, we compare the best-performing SEAL from Table 1, SEAL-dynamic with NCE ranking, with baselines: BERT finetuned with cross-entropy and the state-of-the-art text-MLC model (LACO) (Zhang et al., 2021). Following LACO, we experiment with the Arxiv Academic Paper Dataset (AAPD) (Yang et al., 2018) which consists of computer science papers with annotations of 54 topics. Table 2 shows that SEAL-dynamic with the NCE ranking loss is effective on pre-trained models that utilizes large text datasets as well, surpassing MLC-specific model performance. To efficiently work with large pre-trained models, we further studied utilizing BERT-adapter models on three more text-based datsets (Appendix F) which also showed SEAL-dynamic-NCE model outperforming cross-entropy and InfNet.

Table 2: Test F1 for AAPD

method	**F1** ↑
BERT (cross-entropy)	73.97
LACO (Zhang et al., 2021)	74.9
SEAL-dynamic (NCE)	**75.57**

4.3 Semantic role labeling

To examine SEAL on capturing label dependencies at a sequence level, we perform experiments on semantic role labeling (SRL) (Palmer et al., 2010) using standard benchmark (CoNLL-12) (Pradhan et al., 2013). SRL aims to annotate the predicate-argument structure on plain text extracting *who* did *what* to *whom*. Because the argument roles in a sentence are dependent to each other, injection of

structural priors has shown improvements in performance (Lee et al., 2019; Mehta et al., 2020; Li et al., 2020). To apply SEAL, we utilize pre-trained BERT as the feature network T_E and T_F. To capture the energy on variable-length output, we modify E_Θ^{global} in equation 2 to use self-attention layers. We defer more detailed task description, data statistics, and energy architecture to appendix.

We compare SEAL-dynamic (NCE) on BERT-base against two models: one model trained with cross-entropy loss and another model trained with structured tuning framework (Li et al., 2020) where both models utilize BERT-base with CRF layers (Lafferty et al., 2001). Structured tuning enforces the model to learn the SRL-specific constraint (unique core role) through differentiable constraint loss. Unique core role violation refers to model predicting a type of core argument more than once when it should be unique. To examine effectiveness of SEAL in capturing output dependencies, we report unique-core-role violation rate (UCR) in percentage along with F1 score. While SEAL-dynamic does not enforce any specific constraints nor utilize CRF layers, it shows notable reduction in constraint violation and higher improvement in performance.

Table 3: Results on CoNLL-12. We report F1 and unique core role violation (UCR) in percentage.

	Dev		**Test**	
method	F1 ↑	UCR ↓	F1 ↑	UCR ↓
BERT-CRF	85.92	1.41	85.88	1.32
Structured tuning	86.18	**0.78**	86.09	**0.79**
SEAL-dynamic	**86.99**	0.92	**86.84**	0.96

Table 4: Test mean image IoU for Weizmann Horse segmentation.

method	Mean IoU ↑
FCN (cross-entropy)	77.80± 0.59
DVN with GBI	72.93± 0.52
SEAL-dynamic (regression)	**79.23± 0.35**
SEAL-dynamic (NCE)	76.78± 0.60

4.4 Image segmentation

We evaluate SEAL on binary image segmentation using the Weizmann Horse Image dataset (Borenstein & Ullman, 2004). This task requires a model to learn to predict detailed structures of horse shapes from a scarce 160 training images of a low 32×32 resolution. Following Gygli et al. (2017), we implement Fully Convolutional Network (FCN) (Long et al., 2015) as our feedforward baseline and DVN (Gygli et al., 2017) as our energy network baseline. Given a 3-channel L-pixel RGB image input $\mathbf{x}$, FCN outputs the probability $\tilde{\mathbf{y}} \in [0,1]^L$ that each pixel belongs to a horse object; a binary segmentation mask can be obtained through $\mathbf{1}(\tilde{\mathbf{y}} \geq 0.5)$. In contrast, DVN takes a pair of image $\mathbf{x}$ and soft segmentation mask $\tilde{\mathbf{y}}$ to output their compatibility score and resort to gradient-based inference for prediction. Following Gygli et al. (2017), we train DVN by plugging the soft IoU score

$$s(\tilde{\mathbf{y}}, \mathbf{y}^*) = \frac{\sum_{i=1}^{M} \min(\tilde{y}_i, y_i^*)}{\sum_{i=1}^{M} \max(\tilde{y}_i, y_i^*)}, \quad \text{while IoU}(\mathbf{y}, \mathbf{y}^*) = \frac{\mathbf{y} \cap \mathbf{y}^*}{\mathbf{y} \cup \mathbf{y}^*} \text{ for } \mathbf{y}, \mathbf{y}^* \in \{0,1\}^L, \tag{9}$$

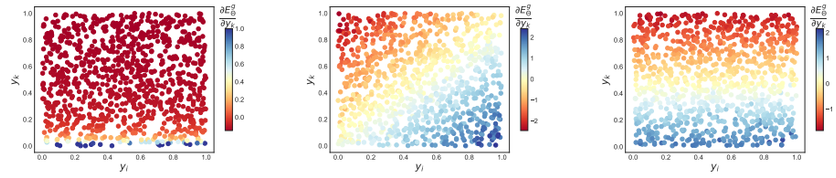
into the Eq.(8). For training with the SEAL framework, we adopt the FCN and DVN architectures for the task-net and the loss-net respectively. In our experiment, we examine SEAL-dynamic with regression which uses the same energy loss as DVN and SEAL-dynamic with the NCE loss because it performs the best in multiple other experiments. More dataset and model details are in Appendix B; experiment details are in Appendix C.2.

Table 4 shows test performances of FCN trained by cross-entropy, DVN, and FCN trained with SEAL-dynamic. Similar to §4.1, we observe that the GBI procedure in DVN training and inference is slow and unstable (Appendix B.3 for more details). While DVN does not outperform feedforward model learned with cross-entropy, the same energy network used as a loss-net in SEAL-dynamic (regression) improves the mean IoU of FCN by 1.43 IoU point and has less variance across random runs. Additionally, we find that SEAL-dynamic with the NCE loss proposed for MLC does not improve FCN. This is likely because many negative samples in NCE are non-realistic as we sample each pixel independently and do not take into account the prior on image that nearby pixels are likely to be in the same class. We leave the exploration of other energy network architectures and sampling strategies in the image domain as future work.

5 Further analysis

In this section, we analyze (1) whether gradient of loss-net rightly captures multivariate dependencies, (2) the effect of ranking loss, and (3) we provide pointers to the data-specific analysis and discussion on the effect of samples in Appendix E.

Visualization of gradients In order to inspect the learning signal provided to the task-net by the loss-net, we plot a single component of the gradient from the global energy of the loss-net, i.e. $\frac{\partial E_{\Theta}^{\text{global}}}{\partial y_k}(y_i, y_k)$. In Figure 2, we inspect the global energy function of DVN trained on the expr_fun, a dataset that provides a taxonomy on the output space. We first plot the discussed gradient component when l_k have positive association with l_i, i.e., $l_i \rightarrow l_k$[4] (2b). Contrasting this with the case when l_k does not have a significant association with l_i (2c), we see that the gradient of the global energy rightly captures the presence and absence of association between the labels. In contrast, binary cross-entropy is incapable of capturing label-to-label association (2a) which is obvious when we inspect its gradient as shown in Appendix A. See Appendix G for more examples and for detailed description of plotting procedure.



(a) BCE w/ and w/o assoc. (b) energy w/ positive assoc. (c) energy w/o significant assoc.

Figure 2: Given there exists a positive association $l_i \rightarrow l_k$ between two labels, the gradient $\frac{\partial E_{\theta}^{\text{global}}}{\partial y_k}$ is plotted (b) and compared with the gradient of BCE (a) when true $l_k = 1$. The gradient of global energy is also plotted (c) for the case when there is no significant association between the two labels.

Effect of applying a ranking loss directly on F_{Φ} With the large improvement that SEAL-NCE achieves, we examine whether the gain is coming from our SEAL framework or simply from the power of a ranking loss itself that can also be applied to the task-net F_{Φ} directly (i.e. $s(\mathbf{x}, \mathbf{y}; \Theta) = \sum_i \log P(\mathbf{y}_i|\mathbf{x}; \Phi_t)$ in equation 8). We conduct this ablation study over relatively small datasets: genbase, cal500, and delicious. While we observe a near +10 F1 point in genbase, we see a decrease in performance on cal500 (-1.0 F1) and delicious (-2.3 F1) compared to the plain cross-entropy model. With these experiments, we first reaffirm that genbase is an outlier dataset where capturing structure can significantly increase the performance. Second, we conclude that the ranking loss without an loss-net is not very effective and that the loss-net indeed plays a major role in the SEAL framework.

6 Conclusion

We propose SEAL, a framework that can adopt a structured energy network as a trainable loss function for training a feedforward network. We show that SEAL is a general framework that can work with various loss functions, such as margin-based, regression-based, and NCE ranking losses, and with different architectures, such as MLPs, CNNs, and pretrained models (BERT). Through extensive experiments on feature-based, text-based MLC, SRL, and image segmentation, we show that SEAL provides an effective way to utilize an energy network and a cross-entropy loss together: performing better, faster, and in a more stable manner when combined. Lastly, through the ablation study, we examine how the loss-net captures the dependencies among output variables. This research opens up doors for various future work, including but not limited to, exploration of different architectures of energy network; application of learned loss-nets in semi-supervised setups. In the long horizon, we are interested in whether general pretrained scoring neural networks, such as BLEURT (Sellam et al., 2020) and BERT-score (Zhang et al., 2019), could serve as a loss function when it can provide backpropagatable gradients to the output space of a feedforward network.

[4]We get label-label dependency from true taxanomy of expr_fun.

References

Aly, R., Remus, S., and Biemann, C. Hierarchical multi-label classification of text with capsule networks. In *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Student Research Workshop*, pp. 323–330, 2019. ISBN 9781950737475. doi: 10.18653/v1/p19-2045.

Belanger, D. and McCallum, A. Structured prediction energy networks. In *33rd International Conference on Machine Learning, ICML 2016*, volume 3, pp. 1545–1554, 2016. ISBN 9781510829008.

Biewald, L. Experiment tracking with weights and biases, 2020. URL `https://www.wandb.com/`. Software available from wandb.com.

Borenstein, E. and Ullman, S. Learning to segment. In *European conference on computer vision*, pp. 315–328. Springer, 2004.

Chen, L.-C., Schwing, A., Yuille, A., and Urtasun, R. Learning deep structured models. In *International Conference on Machine Learning*, pp. 1785–1794. PMLR, 2015.

Devlin, J., Chang, M. W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1:4171–4186, 2019. URL `http://arxiv.org/abs/1810.04805`.

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

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Gutmann, M. and Hirayama, J.-i. Bregman divergence as general framework to estimate unnormalized statistical models. *arXiv preprint arXiv:1202.3727*, 2012.

Gygli, M., Norouzi, M., and Angelova, A. Deep value networks learn to evaluate and iteratively refine structured outputs. In *34th International Conference on Machine Learning, ICML 2017*, volume 3, pp. 2160–2170, 2017. ISBN 9781510855144. URL `https://goo.gl/8OLufh`.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for NLP. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2790–2799. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/houlsby19a.html`.

Huang, C., Zhai, S., Talbott, W., Martin, M. B., Sun, S.-Y., Guestrin, C., and Susskind, J. Addressing the loss-metric mismatch with adaptive loss alignment. In *International Conference on Machine Learning*, pp. 2891–2900. PMLR, 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Lafferty, J., McCallum, A., and Pereira, F. C. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M., and Huang, F. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

Lee, J. Y., Mehta, S. V., Wick, M., Tristan, J.-B., and Carbonell, J. Gradient-Based Inference for Networks with Output Constraints. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:4147–4154, 2019. ISSN 2159-5399. doi: 10.1609/aaai.v33i01.33014147. URL `www.aaai.org`.

Lewis, D. D., Yang, Y., Rose, T. G., and Li, F. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397, 2004.

Li, T., Jawale, P. A., Palmer, M., and Srikumar, V. Structured tuning for semantic role labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8402–8412, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.744. URL https://aclanthology.org/2020.acl-main.744.

Li, Y., Tarlow, D., and Zemel, R. Exploring compositional high order pattern potentials for structured output learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 49–56, 2013.

Long, J., Shelhamer, E., and Darrell, T. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.

Ma, Z. and Collins, M. Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3698–3707, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1405. URL https://aclanthology.org/D18-1405.

Mehta, S. V., Lee, J. Y., and Carbonell, J. Towards semi-supervised learning for deep semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, pp. 4958–4963, 2020. ISBN 9781948087841. URL https://github.com/allenai/allennlp.

Müller, A. C. Methods for learning structured prediction in semantic segmentation of natural images. 2014.

Palmer, M., Gildea, D., and Xue, N. Semantic role labeling. *Synthesis Lectures on Human Language Technologies*, 3(1):1–103, 2010.

Pfeiffer, J., Rücklé, A., Poth, C., Kamath, A., Vulić, I., Ruder, S., Cho, K., and Gurevych, I. Adapterhub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 46–54, 2020.

Pradhan, S., Moschitti, A., Xue, N., Ng, H. T., Björkelund, A., Uryupina, O., Zhang, Y., and Zhong, Z. Towards robust linguistic analysis using ontonotes. In Hockenmaier, J. and Riedel, S. (eds.), *Proceedings of the Seventeenth Conference on Computational Natural Language Learning, CoNLL 2013, Sofia, Bulgaria, August 8-9, 2013*, pp. 143–152. ACL, 2013. URL https://aclanthology.org/W13-3516/.

Rooshenas, A., Zhang, D., Sharma, G., and McCallum, A. Search-guided, lightly-supervised training of structured prediction energy networks. *Advances in Neural Information Processing Systems*, 32: 13522–13532, 2019.

Sandhaus, E. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6 (12):e26752, 2008.

Schwing, A. G. and Urtasun, R. Fully connected deep structured networks. *arXiv preprint arXiv:1503.02351*, 2015.

Sellam, T., Das, D., and Parikh, A. P. Bleurt: Learning robust metrics for text generation. In *Proceedings of ACL*, 2020. URL https://arxiv.org/abs/2004.04696.

Taskar, B., Guestrin, C., and Koller, D. Max-Margin Markov Networks. *Advances in neural information processing systems 16 (2004): 25*, 2004.

Tsochantaridis, I., Hofmann, T., Joachims, T., and Altun, Y. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 104, 2004.

Tu, L. and Gimpel, K. Learning approximate inference networks for structured prediction, 2018. ISSN 23318422.

Tu, L. and Gimpel, K. Benchmarking approximate inference methods for neural structured prediction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 3313–3324, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1335. URL `https://aclanthology.org/N19-1335`.

Tu, L., Pang, R. Y., and Gimpel, K. Improving joint training of inference networks and structured prediction energy networks. In *SPNLP*, 2020.

Vaswani, A. Attention Is All You Need. (Nips), 2017.

Wu, L., Tian, F., Xia, Y., Fan, Y., Qin, T., Jian-Huang, L., and Liu, T.-Y. Learning to Teach with Dynamic Loss Functions. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL `https://papers.nips.cc/paper/2018/hash/8051a3c40561002834e59d566b7430cf-Abstract.html`.

Yang, P., Sun, X., Li, W., Ma, S., Wu, W., and Wang, H. SGM: sequence generation model for multi-label classification. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pp. 3915–3926, 2018.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

Zhang, X., Zhang, Q.-W., Yan, Z., Liu, R., and Cao, Y. Enhancing label correlation feedback in multi-label text classification via multi-task learning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 1190–1200, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.101. URL `https://aclanthology.org/2021.findings-acl.101`.

Checklist

1. For all authors.
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [No] Althgouh we do to some degree, for example, limitation of `SEAL`-dynamic-NCE in image segmentaiton.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] We propose a learning framework which does not have direct negative soceital impacts.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The details are in Appendix.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] The details are in Appendix.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] In the appendix.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]

(b) Did you mention the license of the assets? [Yes]
(c) Did you include any new assets either in the supplemental material or as a URL? [No]
(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Appendix

A Gradient to the label representations

As described in section 4, the energy component of loss L_F, for the task-net, is the sum of following two parts:

$$E_{\Theta}^{\text{local}}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{L} y_i \mathbf{b}_i^{\top} T_E(\mathbf{x}) \quad \text{and} \quad E_{\Theta}^{\text{global}}(\mathbf{x}, \mathbf{y}) = \mathbf{v}^{\top} \gamma(\mathbf{M}\mathbf{y}). \tag{10}$$

The gradient of E^{global} w.r.t $\mathbf{y}$ is given by the following vector of length L

$$\frac{\partial E^{\text{global}}}{\partial \mathbf{y}} = \mathbf{M}^T \text{Diag}(\gamma'(M\mathbf{y}))\mathbf{v}. \tag{11}$$

While the gradient of cross entropy $\text{BCE}(y, y^*)$ w.r.t. $\mathbf{y}$ is given by the following vector of length L

$$\frac{\partial \text{BCE}}{\partial \mathbf{y}} = \mathbf{y}^* \otimes \frac{1}{\mathbf{y}} - (1 - \mathbf{y}^*) \otimes \frac{1}{1 - \mathbf{y}}, \tag{12}$$

where $\otimes$ is element-wise product.

Note the dependency of the gradient for a particular component y_i of y on other component in the expressions above. The gradient of cross entropy ($\frac{\partial l}{\partial y_i}$) only depends on its label dimension i: y_i and y_i^*. In contrast, in the case of the energy function as loss (SEAL), the gradient $\frac{\partial \mathrm{L}_E}{\partial y_i}$ depends on all L dimensions of $\mathbf{y}$ due to the product term $\mathbf{M}\mathbf{y}$ in $\frac{\partial E^{\text{global}}}{\partial y_i}$.

B Task Details

This Section provides task-specific information including task definitions, descriptions for datasets and models, and additional observations in our experiments. Please refer to the beginning of Section 4 for a preliminary description.

B.1 Multi-label classification

Task Definition: Given input x, multi-label classification maps the input to the labels y in the output space, $\mathcal{Y} = \{0, 1\}^L$.

Dataset: In our experiments, we use 11 multi-label classification (MLC) datasets covering various label spaces, label sizes (ranging from 27 to 4k), training sizes, and input characteristics (binary, real, text) as shown in Table 5. The first seven rows are binary- or real-valued feature-based datasets. In addition to including Bibtex and Delicious that were used in baseline approaches (Belanger & McCallum, 2016; Gygli et al., 2017; Tu & Gimpel, 2018), we carefully picked smaller (Genbase, Cal500) and larger datasets (Eurlex-ev) as well as datasets with known taxonomy (Spo_FUN, Expr_FUN) to examine SEAL in various scenarios. For text-based datasets, we select RCV Lewis et al. (2004) and BGC (Aly et al., 2019) which have label dependencies (DAG and forest respectively), AAPD to compare with state-of-the-art text-based MLC model (Zhang et al., 2021), and take a subset of the NYT dataset Sandhaus (2008) [5] to experiment with larger label dimensions and training data. For text-based MLC datasets, we examine whether SEAL can be helpful in the context of large pre-trained models as well. The links to download the multi-label datasets used in this work are given in Table 6.

Feature network: We use standard feed-forward neural network and pretrained BERT-base as the feature network, T_E, for the feature-based and text-based MLC datasets respectively.

[5] Due to their enormous size, we use a subset of NYT training set, which is still the largest dataset in our experiments.

Table 5: Statistics of the datasets used in the experiments.

Dataset	Domain	#Instances		Input Dim	#Labels	Input Type
		Train	Test			
Genbase	Biology	398	132	1186	27	Binary
Cal500	Music	283	114	68	174	Real
Spo FUN	Gene Ontology	1,600	1,266	86	500	Real
Expr FUN	Gene Ontology	1,636	1,288	561	500	Real
Bibtex	Text	4,407	1,497	1836	159	Binary
Delicious	Text	9,690	3,194	500	983	Binary
Eurlex-ev	Text	11,557	3,881	5000	3,993	Real
RCV	Text	13,890	781,265	N/A	104	Raw Text
AAPD	Text	53,840	1000	N/A	54	Raw Text
BGC	Text	58,715	18,394	N/A	142	Raw Text
NYT	Text	175,299	180,659	N/A	2,109	Raw Text

Table 6: Links for downloading the datasets.

Dataset	Link
AAPD	`https://git.uwaterloo.ca/jimmylin/Castor-data/-/tree/master/datasets/AAPD`
BCG	`https://www.inf.uni-hamburg.de/en/inst/ab/lt/resources/data/blurb-genre-collection.html`
NYT	`https://catalog.ldc.upenn.edu/LDC2008T19`
RCV1-V2	`http://www.ai.mit.edu/projects/jmlr/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm`
feature based datasets	`https://www.uco.es/kdis/mllresources/`

Energy network: Given the input $\mathbf{x}$ and the output $y \in \{0,1\}^L$, the energy terms are defined as:

$$E_\Theta^{\text{local}}(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{L} y_i \mathbf{b}_i^\top T_E(\mathbf{x}),\ E_\Theta^{\text{global}}(\mathbf{x},\mathbf{y}) = \mathbf{v}^\top \gamma(\mathbf{M}\mathbf{y}), \tag{13}$$

where $\mathbf{b}_i, \mathbf{v}, \mathbf{M}, T_E$ contain learnable parameters. T_E is a feature network to represent an input $\mathbf{x}$ in the loss-net, and $\gamma(z) = \log(1+e^z)$ is the softplus activation function.

B.2 Semantic Role Labeling

Task Definition: Given a sentence and a predicate pair (W, v), the goal of SRL is to identify the spans of arguments for v in W and assign correct role labels to the identified arguments.

Dataset: We train and evaluate the performance of `SEAL` and baselines on the standard benchmark (CoNLL-12) (Pradhan et al., 2013) for SRL.

Feature network: We use pretrained BERT-base for the feature network, T_E and T_F.

Energy network: For the sequence tagging experiments, we take the input, $\mathbf{x}$, a sequence of N tokens and the output, $\mathbf{y}$, a sequence of N labels. $\mathbf{y}_n$ denotes the label at the nth position in the sequence and has L dimensions, the number of labels. Consequently, we define the energy terms as,

$$E_\Theta^{\text{local}}(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{N}\sum_{j=1}^{L} y_{i,j} \mathbf{b}_j^\top T_E(\mathbf{x})_i,\ E_\Theta^{\text{global}}(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{N} \max E_S(\mathbf{y})_i, \tag{14}$$

where E_S applies the self-attention (Vaswani, 2017) over $\mathbf{y}$ to capture the long-distance dependencies among labels and gets a L by N dimension output. We then do max-pooling over the L dimension before taking the sum across the sequence length to get the global energy, E_Θ^{global}.

B.3 Image segmentation

Task Definition: Given a 3-channel L-pixel RGB image $\mathbf{x}$, binary image segmentation requires models to classify each pixel to either foreground or background, thus outputing a binary segmentation mask $\mathbf{y} \in \{0, 1\}^L$.

Datasets: The Weizmann Horse Image dataset Borenstein & Ullman (2004) consists of 328 left-oriented horse images annotated with segmentation masks. Following CHOPPS (Li et al., 2013) and DVN (Gygli et al., 2017)[6], we adopt a split of 160 training, 40 validation, and 128 testing examples at the resolution of 32×32 pixels. We train models using the training split, select hyperparameters according to mean image IoU on the validation split, and report mean image IoU on the test split.

Feedforward (FCN): We implement Fully Convolutinal Network described by Gygli et al. (2017) as our feedforward baseline and task-net in `SEAL`. It has 3 convolutional layers, 2 transposed convolutional layers, and 2 skip connections. The second transposed convolutional layer outputs two channels of logits. A softmax layer is then applied and we take the normalized horse channel as the probability that each pixel belongs to the horse object. The channel can be binarized to obtain the segmentation mask.

Energy network (DVN): For segmentation, we do not have separate notions of local and global energies (Eq.(2)). We implement Deep Value Network described by Gygli et al. (2017) as our energy network baseline and loss-net in `SEAL`, referring to this public repository. In general we find gradient based inference during DVN training and inference unstable and sensitive to configurations like sampling methods and initialization for the segmentation mask. For example, it achieves < 50 test IoU if we do not supply ground truth samples during its training. Nevertheless, using the same energy network architecture as loss for FCN under the `SEAL`-dynamic learning framework improves the performance of FCN. We leave sampling strategies and loss-net architectures for further exploration.

Qualitative Results: We observe that segmentation masks predicted by FCN trained with `SEAL`-dynamic (regression-s) are better at details (e.g. legs) than those by cross-entropy (Figure 3).

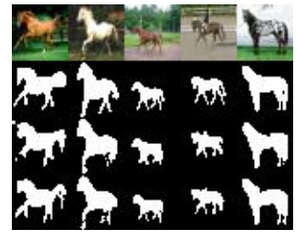


Figure 3: The four rows are the original images, ground truth masks, masks predicted by FCN trained by cross-entropy, and masks by FCN trained with `SEAL`-dynamic.

C Experiment Details

C.1 Training setup

Our experiments are performed on a mix of TitanX, 1080Ti, 2080ti, M40, and RTX8000 GPUs. We use separate ADAM optimizers (Kingma & Ba, 2014) for the loss-net (Θ) and the task-net (Φ) which optimize parameters in alternating fashion[7]. For each minimization loop, we take n_E, n_F gradient steps respectively for estimating Θ_t and Φ_t. The best hyper-parameters for each model are found using Bayesian search[8]. We also report the training time, inference time, and parameter size of different MLC methods in Appendix D.

[6] Script that includes links to and processes the dataset: https://gist.github.com/gyglim/714cb24a1c34c95e0a0c9a8a4ec0620c

[7] The code we used to train and evaluate our models is available a `https://anonymous.4open.science/r/SEAL/README.md`.

[8] We use Weights & Biases (Biewald, 2020) for hyperparameter search

C.2 Hyperparameters

The following tables tabulate the hyper-parameters for SEAL on each dataset. We fix the $\lambda_2 = 1$ in equation 1 to reduce degree of freedom in hyperparameter search. In MLC and SRL experiments, we fix the task-net number of layers, layer-dimensions, and dropout for the SEAL framework according to the best run of the task-net with cross-entropy. In segmentation experiments, we run sweeps of about 200 runs and use the same search space for overlapping hyperparameters for different models for fair comparisons; once the best hyperparameters are found by wandb sweeps, we use them to run with 10 random seeds.

C.2.1 Multilabel Classification

method	n_E	n_F	λ_1	num samples	Θ_{lr}	Φ_{lr}
margin	1	10	0.001	n/a	0.0002	0.001
regression	6	10	0.002	n/a	0.0003	0.001
regression-s	3	10	0.03	20	0.0004	0.001
NCE ranking	9	10	0.002	100	0.0005	0.001

Table 7: genbase

method	n_E	n_F	λ_1	num samples	Θ_{lr}	Φ_{lr}
margin	1	10	0.001	n/a	0.0005	0.008
regression	3	10	1.4	n/a	0.006	0.007
regression-s	9	5	9	5	0.008	0.006
NCE ranking	9	10	0.4	40	0.007	0.005

Table 8: cal500

method	n_E	n_F	λ_1	num samples	Θ_{lr}	Φ_{lr}
margin	1	1	0.0002	n/a	0.002	0.004
regression	3	5	8	n/a	0.0001	0.00015
regression-s	6	1	9	5	0.0001	0.003
NCE ranking	6	5	0.9	20	0.002	0.001

Table 9: delicious

method	n_E	n_F	λ_1	num samples	Θ_{lr}	Φ_{lr}
margin	1	5	0.001	n/a	0.0003	0.002
regression	6	5	0.002	n/a	0.0006	0.004
regression-s	12	1	0.01	20	0.0004	0.007
NCE ranking	6	5	0.5	40	0.001	0.002

Table 10: eurlex

method	n_E	n_F	λ_1	num samples	Θ_{lr}	Φ_{lr}
margin	1	5	0.01	n/a	0.00003	0.001
regression	9	5	0.02	n/a	0.0003	0.0006
regression-s	9	5	2	30	0.007	0.002
NCE ranking	6	10	4	80	0.00005	0.0002

Table 11: expr_fun

method	n_E	n_F	λ_1	num samples	Θ_{lr}	Φ_{lr}
margin	1	10	0.08	n/a	0.0001	0.002
regression	1	5	9	n/a	0.001	0.00015
regression-s	9	10	0.02	10	0.0002	0.0009
NCE ranking	3	10	0.7	60	0.008	0.0005

Table 12: spo_fun

method	n_E	n_F	λ_1	num samples	Θ_{lr}	Φ_{lr}
margin	1	6	0.0002	n/a	0.00005	0.001
regression	9	1	3	n/a	0.005	0.003
regression-s	5	5	9	10	0.005	0.001
NCE ranking	12	5	5	20	0.0004	0.001

Table 13: bibtex

method	n_E	n_F	λ_1	num samples	Θ_{lr}	Φ_{lr}
NCE ranking	1	1	1	300	0.00001	0.0001

Table 14: AAPD

C.2.2 Semantic Role Labeling

We set batch size to 16 and weight decay to 0.00001 for both loss-net (Θ) and the task-net (Φ). For the self-attention used in loss-net (Θ), we set the hidden dimension to 256 and number of attention head to 1. We again fix n_E, n_F as 1.

method	n_E	n_F	λ_1	num samples	Θ_{lr}	Φ_{lr}
NCE	1	1	0.5	400	0.00001	0.00001

Table 15: CoNLL-12

C.2.3 Weizmann Horse Segmentation

In all experiments, we use batch size 8 and 36 image crops (Gygli et al., 2017). In DVN with GBI, the found learning rate for GBI to get adversarial samples is 7.82 while that for other GBI is 6.96.

method	n_E	n_F	λ_1	num samples	Θ_{lr}	Θ_{wd}	Φ_{lr}	Φ_{wd}
FCN (cross-entropy)	n/a	n/a	n/a	n/a	n/a	n/a	2.01e-3	5.86e-5
DVN with GBI	n/a	n/a	n/a	n/a	2.73e-4	4.73e-4	n/a	n/a
SEAL-dynamic (regression)	8	10	0.20	n/a	1.9e-2	1.41e-4	3.64e-4	3.75e-5
SEAL-dynamic (NCE)	1	1	0.0002	500	1.01e-5	4.17e-4	3.96e-3	3.59e-5

Table 16: Weizmann Horse segmentation. lr is learning rate. wd is weight decay.

D Analysis: Parameter Size and Speed of Inference and Training

We present the analysis on number of parameters (Table 17), inference speed (Table 18), and training speed (Table 19) that different methods utilize for Multilabel Classification datasets.

In summary, SEAL requires approximately twice number of parameters compared to cross-entropy loss on training time, however it requires the same amount of parameter on inference time. Similarly,

train time (sec) per epoch of SEAL is slower than that of cross-entropy and energy network as SEAL requires multiple backpropagation steps for both energy network and feedforward network. Nonetheless, again, at inference time, the speed of CE and SEAL is the same and is much faster ($2\times$-$7\times$) than the inference of energy network.

The analysis we present below are on the seven feature-based dataset, however the trend is similar for text-based dataset as well: twice parameter size on training time but equal parameter and runtime in inference time with CE method. The only difference in the text-based MLC is, due to running on a much larger BERT-baed models, that we only run one step for each update of feedforward network and energy network. Thus, unlike Table 19, train time of SEAL is approximately twice that of CE.

Dataset \ Methods	Parameter size			
	CE	Energy network (SPEN, DVN, NCE)	SEAL (Train-time)	SEAL (Inference-time)
EXPR_FUN	333000	533800	866800	333000
SPO_FUN	447000	597600	1044600	447000
Bibtex	958800	991000	1949800	958800
Cal500	1158700	1123500	2282200	1158700
Delicious	754000	951000	1705000	754000
Genbase	485600	491400	977000	485600
Eurlex-ev	4747500	5546500	10294000	4747500

Table 17: The number of parameters required during train time for SEAL is approximately double the size of feedforward (CE column) while energy network and feedforward sizes are comparable. However, in the inference time, SEAL has an equal amount of parameters to the CE column as only feedforward network is utilized during inference time.

Dataset	Inference time (sec)		Inference speed (examples/sec)		Speed ratio (CE and SEAL/ Energy Network)
	Energy network	CE and SEAL	Energy network	CE and SEAL	
EXPR_FUN	1.33	0.22	638	3801	5.96
SPO_FUN	1.18	0.16	709	5231	7.38
Bibtex	3.6	1.78	414	840	2.03
Cal500	0.24	0.10	438	1080	2.47
Delicious	5.35	1.39	599	2307	3.85
Genbase	0.23	0.13	574	1005	1.75
Eurlex-ev	24	12.22	162	317	1.96

Table 18: We simply average inference time for CE and SEAL variants as they are very similar. Likewise, we average the inference time of different energy networks. Here, inference time (sec) is recorded for the whole validation set. We also present speed per sec (example/sec) and speed ratio. The last column shows that CE and SEAL methods are 2x-7x faster in inference time than the energy networks.

Methods\ Datasets	Train time (sec/epoch)						
	EXPRFUN	SPO_FUN	Bibtex	Cal500	Delicious	Genbase	Eurlex-ev
CE	6.29	5.46	22.67	1.00	33.01	2.12	114.84
SPEN	10.99	11.01	28.04	2.53	37.31	3.57	136.10
DVN	10.95	10.02	32.12	1.87	55.24	2.79	129.94
NCE	3.71	3.83	13.97	2.82	22.03	3.68	89.33
SEAL-margin	27.96	35.24	212.94	5.78	44.60	8.73	260.10
SEAL-regression	46.32	56.97	27.78	8.07	143.90	13.91	352.63
SEAL-regression-s	42.87	77.92	179.97	8.48	45.33	10.79	221.33
SEAL-NCE	72.43	40.34	131.73	16.58	158.97	17.77	431.92
SEAL-Ranking	41.24	26.83	218.37	7.04	317.63	10.74	408.79

Table 19: The training time per epoch is presented per dataset and per loss function used. Due to different gpu types and node status, there are some outliers. Furthermore, as SEAL runs multiple number of backpropagation steps for energy network and feedforward network in the alternating optimization, direct comparison of training time is not really available. However, the general trend in terms of training time per epoch is CE $<$ Energy Networks $<$ SEAL.

E Further Analysis

E.1 Effect of samples

Given the notable performance of the NCE ranking loss, we raise the question of whether drawing many samples from the task-net helps better estimation of the energy surface. To examine this question, we perform additional experiments using the regression loss with sampling. Given a sample set S, we define the regression-s loss as $\mathrm{L}_{E-\text{regression-s}} = \sum_{\tilde{\mathbf{y}} \in S} \mathrm{L}_{E-\text{regression}}(\mathbf{x}, \mathbf{y}^*, \tilde{\mathbf{y}}; \Theta)$. We take two approaches in collecting these samples: (1) discrete binary vectors drawn from probability vector $\tilde{\mathbf{y}}$ as done with the NCE ranking loss and (2) continuous perturbation of $\tilde{\mathbf{y}}$ with Gaussian noise. We find that discrete samples do not make any notable changes between regression and regression-s, while continuous samples result in +0.4 F1 improvements on average. We report results of regression-s using continuous samples in row 10 of Table 1. We conclude that sampling helps capturing better energy surfaces for training the task-net. However, the effect and characteristics of the samples might differ for different types of energy losses. We believe sampling is more effective for the NCE ranking loss where the group of samples contribute in a relative manner in learning surface, in contrast to the regression-s loss to which each sample contributes independently.

E.2 Data-specific analysis

We analyze the characteristics of selected datasets to reason about the high performances of `SEAL` on them.

E.2.1 genbase

Here, we analyze the factors behind `SEAL` achieving almost perfect F1 score on the genbase dataset. It seems genbase stands out with its small label size (27 which is the smallest among Table 5) and with very clear pattern in the label space. Upon analysis, out of 27 labels, we found that 6 labels only occur as a singleton (by itself), 10 labels only occur as non-singleton, and 7 labels do not participate at all. This leaves only 4 labels occurring by themselves as well as with others. Not only that, only 7% (35/500) of training instances have more than one active label. In this peculiar setting, we conjecture feedforward networks with cross-entropy (CE) loss can easily learn singletons, and energy networks would easily learn co-occurrence, but learning both might be confusing for these two models at the opposite end of the spectrum. We believe the synergy of CE and a loss from the energy network enables capturing the best of both worlds and achieving a near-perfect score that none of the approaches by itself achieves. As can seen in Table 1, neither cross-entropy nor energy networks are nearly as strong as SEAL methods.

E.2.2 Delicious and Cal500

In Table 1, it seems `SEAL`-NCE and `SEAL`-Ranking are particularly well performing on cal500 and delicious. It turns out that cal500 and delicious have a very high diversity of 1 and 0.981 (Reference:

MLC data repo). Diversity of 1 means that each data point holds a unique label set. It seems that the exposure to multiple samples and evaluations of their relative scores in the NCE ranking loss function can be helpful in a high-diversity setting.

F Additional Experiment: Multi-label Classification on Text Dataset with BERT Adapters

Table 20: Test F1 for text MLC datasets.

method \ datasets	**BGC**	**RCV**	**NYT**
cross-entropy	81.15	87.18	77.4
SEAL-dynamic with margin (InfNet)	81.14	87.01	78.13
SEAL-dynamic with NCE	**81.64**	**87.82**	**78.87**

To test the effectiveness of SEAL on training large language models, in Table 20, we further experiment with text datasets using pre-trained BERT (Devlin et al., 2019) with adapter (Houlsby et al., 2019; Pfeiffer et al., 2020) as the feature network T_E and T_F. Considering the computational expense of running hyper-parameter searches on BERT, based on Table 1, we choose the best-performing SEAL (SEAL-dynamic with NCE ranking) to compare with baselines: InfNet and cross-entropy. Table 20 shows that SEAL-dynamic with the NCE ranking loss is effective on pre-trained models that utilizes large text datasets as well. On average, SEAL-NCE gains 0.87 and 0.68 F1 points over cross entropy and SEAL-margin (InfNet) respectively on the text datasets.

G Gradient Plots

In multi-label classification, the global energy $E_\Theta^{\text{global}}(\mathbf{y})$, one of the two constituents of the loss-net, captures the dependence between different labels. In order to inspect the nature of the dependence captured, we randomly pick a tuple of labels (l_i, l_k) such that $l_i \bowtie l_k$, where $\bowtie$ is a particular relation and plot samples of $\frac{\partial E^{\text{global}}}{\partial y_k}(\mathbf{y})$ in the following manner. First, we sample each component of $\mathbf{y}$ uniformly and independently, which, with a slight abuse of notation can be written as $\mathbf{y} \sim \text{Uniform}(0,1)^L$; then heatmap of $\frac{\partial E_\Theta^{\text{global}}(\mathbf{y})}{\partial y_k}$ on coordinates (y_i, y_k) are plotted. Figure 2b (see Fig. 4, 6 for more examples) shows the samples from $\frac{\partial E^{\text{global}}}{\partial y_k}(y_i, y_k)$ when i and k are such that $y_i \implies y_k$, i.e. there is a positive association between labels l_i and l_k. As seen, larger values of y_i makes the gradient w.r.t y_k always higher for the same value of y_k showing positive association between y_i and y_k. Contrast this with Figure 2c (see Fig. 5, 7 for more examples), where i and k are such that there is no relationship between y_i and y_k. In this setting, $\frac{\partial E_\Theta^{\text{global}}}{\partial y_k}$ does not change w.r.t y_i. Furthermore, as expected, Figure 2a, shows that when the same procedure is applied to the cross-entropy loss function, we find no dependence between any pair of labels. For the plots discussed in this section, we analyze energy model trained on DVN (4, 5) and SEAL-dynamic-NCE (6, 7) trained on the expr_fun dataset.

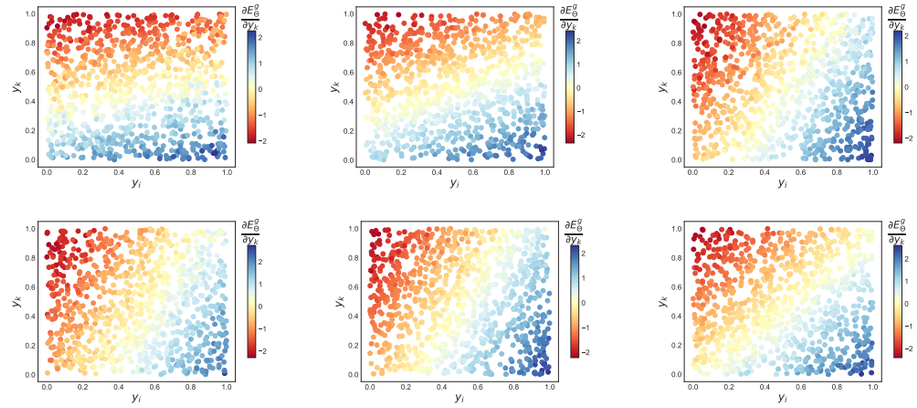


Figure 4: $(y_i,\ \frac{\partial E_{\Theta}^{global}(\mathbf{y})}{\partial y_k})$ when $y_i \implies y_k$. Analysis of energy model trained with DVN.

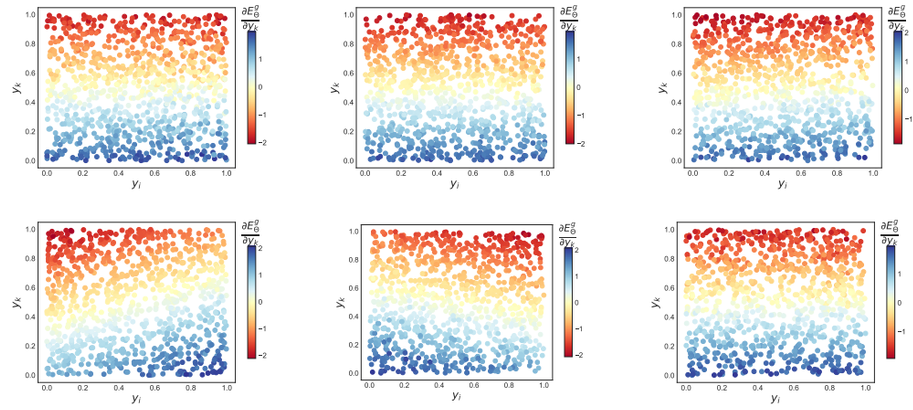


Figure 5: $(y_i,\ \frac{\partial E_{\Theta}^{global}(\mathbf{y})}{\partial y_k})$ when there is no relation between labels l_i and l_k. Analysis of energy model trained with DVN.

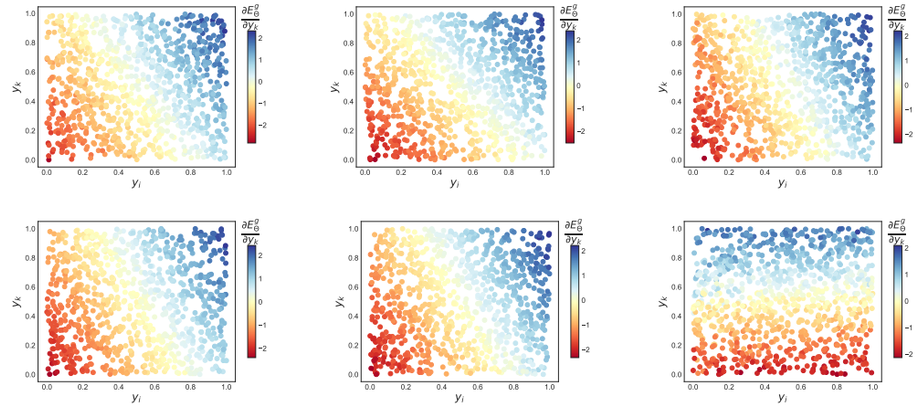


Figure 6: $(y_i,\ \frac{\partial E_{\Theta}^{global}(\mathbf{y})}{\partial y_k})$ when $y_i \implies y_k$. Analysis of energy model trained with SEAL-dynamic NCE.

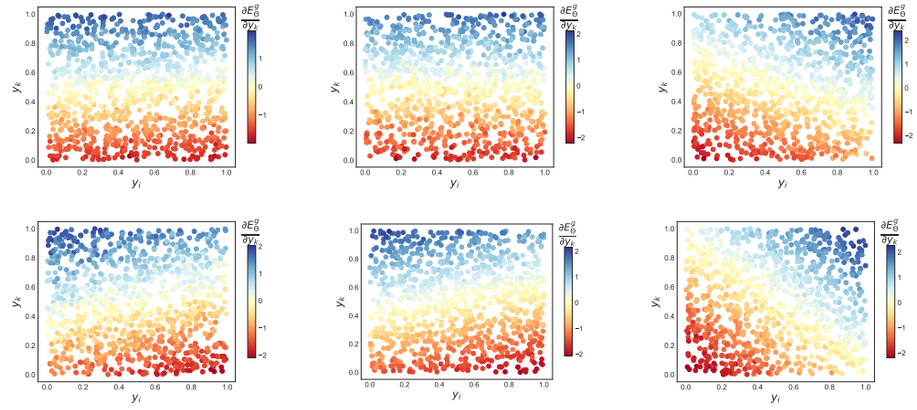


Figure 7: $(y_i,\ \frac{\partial E_{\Theta}^{global}(\mathbf{y})}{\partial y_k})$ when there is no relation between labels l_i and l_k. Analysis of energy model trained with SEAL-dynamic NCE.