Modeling Global Factors in Probabilistic Structured Neural Models

Kartik Goyal

CMU-LTI-19-002

October 23, 2019

School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:
Chris Dyer *(chair)*
Taylor Berg-Kirkpatrick *(chair)*
Graham Neubig
Alexander Rush

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in Language and Information Technologies.

Abstract

Neural sequence models have been successful at modelling rich and structured outputs associated with combinatorial objects likes sentences, paragraphs, sets etc which in turn have been used for several important tasks like machine translation, summarization etc. Different classes of neural models have their own strengths and weaknesses along several axes like generalizability and recognition of inductive bias, ease of optimization, awareness of structure, interpretability, computational complexity and many others. This thesis, in particular, focuses on analyzing and ameliorating issues relating to training of two broad classes of neural sequence models:

1. *Locally normalized* models which typically use chain rule in a left-to-right manner to express the probability of a sequence and hence involve explicit normalization at each step in the inference.
2. *Globally normalized* models that abandon the chain rule and involve approximation of statistics that require explicit enumeration over all of the hypothesis space.[1]

Locally normalized models are prevalent because of ease of computation of the probability using chain rule. However, a common algorithm, *teacher forcing*, for training these models suffers from *exposure bias* i.e. the training is not exposed to model's own decoding errors made at previous decoding steps. We present two solutions to this issue so that the training of the models is more *search-aware*. These approaches involve continuous relaxations to the commonly used discontinuous decoding procedures with neural sequence models. One of the solutions is inspired from the *scheduled sampling* procedure and extends it by relaxing the procedure via temperature-based softmax such that the objective is differentiable and aware of discontinuous greedy decoding or sampling performed during inference. Another solution focuses on making the training procedure aware of beam search as the inference procedure of choice during decoding. This method proposes a continuous objective function which involves a continuous relaxation of the beam search procedure. The continuous relaxation of the discontinuous decoding procedures results in successful incorporation of the decoding algorithms into backpropagation-based training which requires the presence of gradients or subgradients. These approaches result in significant improvements over teacher-forcing for tasks like Machine Translation, Named Entity Recognition, and CCG supertagging.

We also present a method to train *globally normalized* sequence models, in which we modify the above-mentioned search-aware algorithm involving continuous relaxation of the beam search and find that in the context of inexact inference using beam search, training with the globally normalized strategy results in models that are more effective at responding to search errors during training compared to their locally normalized counterparts. This is in spite of the fact that assuming flexible neural parametrization, locally normalized models are theoretically as expressive as globally normalized models. We attribute this finding to globally normalized models ameliorating the issue of *label*

[1] A common assumption is that the hypothesis space or the total number of sequences are countable.

bias commonly associated with locally normalized models. One of the contributions in this work is a reinterpretation of *label bias* that lets us explain the potential reason for amenability of globally normalized sequence models to search-aware training in context of inexact search.

Finally, this thesis proposes to view training of structured *globally normalized* models as minimization of *energy* in an energy-based interpretation of the globally normalized models. More specifically, we propose algorithms for sequences that, instead of building scores in left-to-right stepwise manner, focus on scoring chunks of random variables at once. A lot of recent works have speculated based upon empirical evidence,that stepwise nature of score construction might be the main cause of calibration issues, favoring of ill-formed sentences, repetition and other problems associated with neural sequence models. Hence, this thesis focuses on developing better globally-normalized models involving flexible energy networks for density estimation of sequences that abandon the left-to-right generation constraint. Aside from ameliorating the above-mentioned issues, these models will also allow for massive parallelization for faster decoding. From the perspective of computational efficiency, I also propose to explore approaches that would let the energy network adapt its depth according to the complexity of the input instance by modelling the depth-wise transformation of hidden states of a neural energy network as continuous evolution of a neurally parametrized differential equation.

Additionally, we propose algorithms to train these flexible energy networks to model objects belonging to domains of disparate nature that are different from the domain of discrete sequences. This will enable study of the effectiveness of global normalization for problems of varied nature. Specifically, the two domains we propose to study are: i) **Images**: that have a continuous domain and hence are more amenable to gradient based methods for inference, and ii) **Discrete Sets**: that have a discrete domain like sequences but the crucial difference is that the models for these objects need to take permutation invariance into account due to inherent orderlessness among the members of a set.

Chapter 1

Introduction

A probabilistic framework for statistical models is useful not only for predictive tasks but also for characterizing uncertainty, incorporating prior knowledge, and obtaining expectation-based statistics that consider all the possible outcomes in a principled manner. Moreover, characterizing probability densities associated with these models using universal function approximators like deep neural networks allow for learning extremely expressive and flexible models that can account for observations in varied modalities.

This thesis addresses the kinds of probabilistic neural models that have several interdependent random variables, typically discrete, all of which jointly comprise a large space of possible outcomes. Both training and inference rely on efficient exploration of the hypothesis space that grows exponentially with the number of random variables in the model. In particular, this thesis focuses heavily on models for sequence prediction. The interdependence between the random variables is often characterized by some structural assumptions on the organization of the random variables and hence from hereon, the output spaces associated with these models will be referred as *structured output spaces* and these models themselves will be referred to as *structured prediction models* when used for prediction.

A natural method of inference in these models is to use the chain rule over the random variables following some ordering such that the computation is minimized. For sequence models, the typical arrangement is left-to-right with the chain rule conditioning each random variable over its left history. We refer to the models that have been trained using this method as *locally normalized* models. Inference in these models involves a normalization operation at each decoding step and is carried out sequentially.

Another strategy with a finite number of random variables would be to abandon the chain rule and associate each configuration of the random variables with a non-negative score. For finite length sequences then, a probability distribution function can be defined which involves a normalization constant that is the sum of scores of each possible sequence. Models trained with this strategy are referred to as *globally normalized* models. A major challenge in training the globally normalized models is computing the normalizing constant which involves enumeration over all of the hypothesis space.

In this thesis we explore the challenges associated with training both these classes of models and devise solutions to counter these challenges. For sequences, locally normalized models are

a prevalent class of models because of ease of computation of the probability using chain rule. However, the most common algorithm for training these models (Sutskever et al., 2014) suffers from *exposure bias* i.e. the training is not exposed to model's own decoding errors made at previous decoding steps. We present two solutions to this issue so that the training of the models is *search-aware*. In the context of training neural sequence models, it is desirable that the objective be a continuous function of the parameters such that it can simply be implemented as a computation graph and the end-to-end training be performed via backpropagation. Hence, in this thesis we devise and experiment with *search-aware* algorithms for training that are also continuous with respect to the model parameters. One of the solutions (Goyal et al., 2017a) is inspired from the *scheduled sampling* (Bengio et al., 2015) procedure and extends it by relaxing the procedure such that the objective is differentiable. Another solution (Goyal et al., 2018) is presented to train in a search-aware manner with a continuous objective function which involves a continuous relaxation of the beam search procedure so that the training procedure accounts for inference with beam search. These approaches result in significant improvements overteacher-forcing for tasks like Machine Translation, Named Entity Recognition, and CCG supertagging.

This thesis also explores *globally normalized* models for problems related to sequence prediction. We modify the search-aware algorithm involving continuous relaxation of the beam search to train globally normalized sequence models and find (Goyal et al., 2019) that in the context of inexact inference using beam search, training with the globally normalized strategy results in models that more effective at responding to search errors during training compared to their locally normalized counterparts. This is in spite of the fact that assuming flexible neural parametrization, locally normalized models are theoretically as expressive as globally normalized models. We attribute this finding to globally normalized models ameliorating the issue of *label bias* commonly associated with locally normalized models. One of the contributions in this work is a fresh reinterpretation of *label bias* that lets us explain the potential reason for amenability of globally normalized models to search-aware training in context of inexact search.

Finally, this document proposes to view training of structured *globally normalized* models as minimization of *energy* (LeCun et al., 2006) in an energy-based interpretation of the globally normalized models. More specifically, we propose algorithms for sequences that, instead of being iterative over steps of the sequence, focus on scoring chunks of random variables at once. It is important to abandon the left-to-right score building paradigm because a lot of recent work (Holtzman et al., 2019; Welleck et al., 2019b) focused on diagnosing neural sequence models speculates that the left-to-right generation paradigm might be the main cause of calibration issues, repetition of words, assignment of high scores to ill-formed sequences and many other issues associated with neural sequence models. The hypothesis this document proposes is that if the score construction of sequences are aware of the global factors pertaining to the whole sequence, then the subsequent globally normalized model will result in globally comprehensive and well behaved sequences. Aside from ameliorating the above-mentioned issues, these global models will also allow for massive parallelization as a virtue of having abandoned the sequential score construction paradigm for faster decoding. From the perspective of computational efficiency, I also propose to explore approaches that would let the energy network adapt its depth according to the complexity of the input instance by modelling the depth-wise transformation of hidden states of a neural energy

network as continuous evolution of a neurally parametrized differential equation (Chen et al., 2018).

In order to study optimization challenges involved with and develop robust globally normalized models for sequences, this thesis also proposes to develop flexible globally normalized models for objects belonging to domains of disparate nature which are related yet different from domain of discrete sequences. The two other domains we propose to study are:

1. **Images**: They have a naturally continuous and ordered representation. Hence, in many ways, they are suitable for modelling via Energy networks which readily define a mapping on continuous domains. This also lets us develop gradient-based methods for inference and inference-aware training in energy networks. This family of generative models for images is relatively underexplored while other common kinds of generative models like autoregressive models, VAEs, and GAN based generative models suffer from one or more deficiencies like lack of an explicit likelihood, poor likelihood maximization, poor generation of samples, computational inefficiencies etc. We believe that energy based models will ameliorate most of these deficiencies.
2. **Discrete Sets**: Like sequences, they are discrete and defining a mapping over continuous energy domains is not as straightforward as it is for images. This also precludes gradient based inference without a continuous relaxation to representation of sets. However, being countable (and finite for application considered in this document), they provide an opportunity for more efficient approximation to enumeration over hypothesis space and hence a better approximation to the partition function. A crucial difference from sequences is that the energy models for sets need to be invariant to permutations of elements in the sets due inherent orderlessness among members of a set.

To summarize, this document describes our work on search-aware training methods that are amenable to backpropagation for locally normalized neural sequence models that ameliorate the issues related to other training methods like teacher forcing. Furthermore, this work also describes work on training globally normalized models for sequences which are demonstrated to be superior to their comparable locally normalized counterparts. Finally, this document proposes to develop better globally normalized models via an energy network based parametrization, that are flexible, robust and computationally efficient for a variety of domains including discrete sequences, images and discrete sets.

1.1 General Notation

In this document, the input observation to these models is commonly denoted by $\mathbf{x}$ and the output sequence is denoted by $\mathbf{y}$ unless specified otherwise. The output space of all the possible outcome spaces is denoted by $\mathcal{Y}$. The output from the model is denoted by $\hat{\mathbf{y}}$ and the target output in annotated data is denoted by $\mathbf{y}^*$. The individual random variables in a structured model are denoted by an index in the subscript and subsets of random variables are denoted by a range of indices in the subscript. We use $s(\mathbf{x}, \mathbf{y}_{1:i-1}, \mathbf{y}_i)$ to refer to a non negative score of output label y at time-step i for the input $\mathbf{x}$ and the prediction history $\mathbf{y}_{1:i-1}$ and let V be the label space.

1.2 Locally Normalized Sequence Models

For modeling sequences using neural networks, this class of models is the most prevalent in literature. Under a locally normalized sequence model $\mathcal{M}_L$, the probability of $\mathbf{y}$ given $\mathbf{x}$ is:

$$p_{\mathcal{M}_L}(\mathbf{y} \mid \mathbf{x}) = \prod_{i=1}^{n} p(\mathbf{y}_i \mid \mathbf{x}, \mathbf{y}_{1:i-1}) = \prod_{i=1}^{n} \frac{s(\mathbf{x}, \mathbf{y}_{1:i-1}, \mathbf{y}_i)}{Z_{L,i}(\mathbf{x}, \mathbf{y}_{1:i-1})}$$

where $Z_{L,i}(\mathbf{x}, \mathbf{y}_{1:i-1}) = \sum_{y \in V} s(\mathbf{x}, \mathbf{y}_{1:i-1}, y)$, is the local normalizer at each time step and n is the number of prediction steps. Since, the local normalizer is easy to compute, likelihood maximization based training is a standard approach for training these models. As mentioned earlier, the typical method for training locally normalized is not *search-aware* and suffers from *exposure bias*. This thesis explores two solutions to make the training search-aware while keeping the whole training procedure end-to-end (sub)-differentiable.

1.3 Globally Normalized Sequence models

In contrast, under a globally normalized model $\mathcal{M}_G$, the probability of $\mathbf{y}$ given $\mathbf{x}$ is:

$$p_{\mathcal{M}_G}(\mathbf{y} \mid \mathbf{x}) = \frac{\prod_{i=1}^{n} s(\mathbf{x}, \mathbf{y}_{1:i-1}, y_i)}{Z_G(\mathbf{x})}$$

where $Z_G(\mathbf{x}) = \sum_{\mathbf{y} \in \mathcal{Y}} \prod_{i=1}^{n} s(\mathbf{x}, \mathbf{y}_{1:i-1}, \mathbf{y}_i)$, is the global log-normalizer. $Z_G(\mathbf{x})$ is intractable to estimate for most problems of interest due to the large search space therefore, an exact likelihood maximization training approach is intractable for these models and we have to rely on inexact search. In our experiments, we show that globally normalized models are more amenable to recover from search errors during training in the context of inexact search and also ameliorate the issues related to *label bias* commonly encountered with locally normalized models. Also, these models naturally lend to computation of marginal probabilities of subsets of nodes in a sequence and hence are more amenable to train with external constraints on the posterior distribution.

Chapter 2

Training Neural Sequence Models: Ameliorating Exposure Bias

In this chapter, we focus on training of search-aware locally normalized neural sequence models. We identify that a common cross-entropy optimization algorithm for training locally normalized sequence models suffers from *exposure bias* and propose objective functions that take the model's predictions into account for ameliorating this bias while maintaining end-to-end (sub)-differentiability of the objective so that training can be performed via automatic differentiation.

2.1 Standard Cross-entropy training

A common method to train neural sequence models is maximizing likelihood of target sequences $\mathbf{y}^*$, conditioned on the input sequence $\mathbf{x}$ in a locally normalized model. The input is encoded via an expressive neural parametrization, typically a bidirectional LSTM to get an encoded representation of the input. A neural recurrent decoder then generates the output sequence in a left-to-right manner with the log probability of the sequence computed using chain-rule as

$$\log p(\mathbf{y}^* \mid \mathbf{x}) = \sum_{i=1}^{T} \log p(\mathbf{y}_i^* \mid \mathbf{y}_{1:i-1}^*, \mathbf{x}) \tag{2.1}$$

It is important to notice that in the formulation above, the probability of the target label at a particular decoding step is conditioned on the gold target history. This means that in recurrent decoder like an LSTM, the dynamics of hiddden states is guided by assuming access to the gold target history. Another possible parametrization for training could be to condition the probability of the gold target on each step on the input sequence and the model 's predicted history i.e.:

$$\log p(\mathbf{y}^* \mid \mathbf{x}) = \sum_{i=1}^{T} \log p(\mathbf{y}_i^* \mid \hat{\mathbf{y}}_{1:i-1}, \mathbf{x}) \tag{2.2}$$

Equation 2.1 is the standard method to train the locally normalized model because it is naturally end-to-end differentiable and has shown to result in well-trained discriminative sequence models.

Equation 2.2 reasons about model's behavior over its possible predictions in past while deciding about current prediction. We call the objective in Equation 2.2 *search-aware* because it is aware of inference with the model. As we discus in the following sections, search-aware training poses some challenges for training related to differentiability and tractability but it also has potential to result in more robust and well-informed models.

The objectives above, are similar to objective of well-known *Maximum Entropy Markov Models*(MEMMs)(**?**), which focus on optimizing an independent classification loss at each decode-step conditioning the prediction on the input $\mathbf{x}$ and the history $\mathbf{y}_{<}$. The neural LSTM parametrization allows us to condition on *full* input and *complete* history, which makes these models very expressive.

2.2 Exposure Bias

The standard cross-entropy training procedure described above always uses gold contexts instead of model's predicted context during training of the recurrent parameters of the decoder. Hence, the states and contexts encountered during training do not match those encountered at decode time and it is likely for the decoder to encounter recurrent hidden states that it never encountered during training making it difficult to recover from errors in model's predictions during decoding. This phenomenon of mismatch between training and decoding phases is referred to as *exposure bias*(Ranzato et al., 2016). It has been observed that the models that use a more advanced inference procedure than greedy decoding like beam search during decoding, but do not account for it during training sometimes yield reduced test performance when compared with greedy decoding (Koehn and Knowles, 2017; Neubig, 2017; Cho et al., 2014). This issue has been addressed using several approaches that try to incorporate awareness of decoding choices into the training optimization. These include reinforcement learning (Ranzato et al., 2016; Bahdanau et al., 2017), imitation learning (Daumé et al., 2009; Ross et al., 2011; Bengio et al., 2015), beam-search based approaches (Wiseman and Rush, 2016; Andor et al., 2016; Daumé III and Marcu, 2005) and other approaches that focus on decoding procedures based on variational approximation (Gormley et al., 2015). *scheduled sampling* (Bengio et al., 2015) is another simple approach designed to counter exposure bias which stochastically incorporates contexts from previous decoding decisions into training.

In this chapter, we propose two solutions to ameliorate exposure bias. The first solution is inspired from a technique called 'scheduled sampling' which involves mixing in models predictions instead of the gold labels at random decoding steps to be fed in as the input embedding at the next step. This approach while shown to work has a major flaw in that it makes the objective discontinuous wrt. the model parameters and hence not amenable for end-to-end training via an automatic differentiation toolkit. We present an algorithm that is end-to-end differentiable and is inspired by scheduled sampling. The second solution involves making the model aware to beam search while training. Because the outputs of the sequence models are discrete and the beam search procedure, commonly used during inference, is itself discontinuous, there is no straightforward way to incorporate beam search into training in an end-to-end (sub)-differentiable manner. We propose a continuous relaxation to the beam search procedure and such that we can interpret the beam-search aware objective as a computation graph and train the model using automatic differentiation libraries. We perform experiments on Machine Translation(MT) and tagging tasks like Named

entity recognition and CCG supertagging to exhibit the empirical superiority of our search-aware solutions over standard cross-entropy based training baselines.

2.3 Differentiable Scheduled Sampling

One the simplest to implement and least computationally expensive approaches to ameliorate exposure bias is *scheduled sampling* (Bengio et al., 2015), which stochastically incorporates contexts from previous decoding decisions into training. While scheduled sampling has been empirically successful, its training objective has a drawback: because the procedure directly incorporates greedy decisions at each time step, the objective is discontinuous at parameter settings where previous decisions change their value. As a result, gradients near these points are non-informative and scheduled sampling has difficulty assigning credit for errors. In particular, the gradient does not provide information useful in distinguishing between local errors without future consequences and cascading errors which are more serious.

Hence in this thesis, a novel approach based on scheduled sampling is proposed that uses a differentiable approximation of previous greedy decoding decisions inside the training objective by incorporating a continuous relaxation of argmax. As a result, our end-to-end relaxed greedy training objective is differentiable everywhere and fully continuous. By making the objective continuous at points where previous decisions change value, this approach provides gradients that can respond to cascading errors. In addition, we demonstrate a related approximation and reparametrization for sample-based training (another training scenario considered by scheduled sampling (Bengio et al., 2015)) that can yield stochastic gradients with lower variance than in standard scheduled sampling. In the experiments on two different tasks, machine translation (MT) and named entity recognition (NER), we show that our approach outperforms both cross-entropy training and standard scheduled sampling procedures with greedy and sampled-based training.

2.3.1 Discontinuity in Scheduled Sampling

While scheduled sampling (Bengio et al., 2015) is an effective way to rectify exposure bias, it cannot differentiate between cascading errors, which can lead to a sequence of bad decisions, and local errors, which have more benign effects. Specifically, scheduled sampling focuses on learning optimal behavior in the current step given the fixed decoding decision of the previous step. If a previous bad decision is largely responsible for the current error, the training procedure has difficulty adjusting the parameters accordingly. The following machine translation example highlights this credit assignment issue:

Ref: The cat purrs . Pred: The dog barks .

At step 3, the model prefers the word ‘barks’ after incorrectly predicting ‘dog’ at step 2. To correct this error, the scheduled sampling procedure would increase the score of ‘purrs’ at step 3, conditioned on the fact that the model predicted (incorrectly) ‘dog’ at step 2, which is not the ideal learning behaviour. Ideally, the model should be able to backpropagate the error from step 3 to the source of the problem which occurred at step 2, where ‘dog’ was predicted instead of

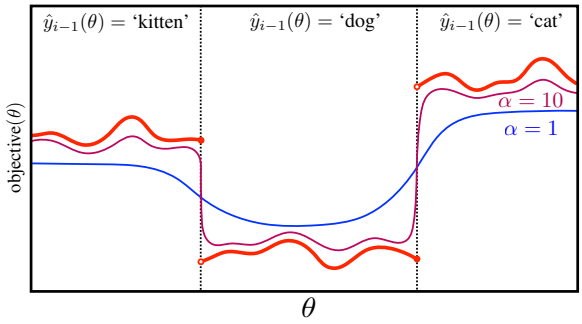

Figure 2.1: Discontinuous scheduled sampling objective (red) and continuous relaxations (blue and purple).

'cat'. The lack of credit assignment during training is a result of discontinuity in the objective function used by scheduled sampling, as illustrated in Figure 2.1. We denote the ground truth target symbol at step i by y_i^*, the embedding representation of word y by $e(y)$, and the hidden state of a seq2seq decoder at step i as h_i. Standard cross-entropy training defines the loss at each step to be $\log p(y_i^*|h_i(e(y_{i-1}^*), h_{i-1}))$, while scheduled sampling uses loss $\log p(y_i^*|h_i(e(\hat{y}_{i-1}), h_{i-1}))$, where $\hat{y}_{i-1}$ refers the model's prediction at the previous step.[1] Here, the model prediction $\hat{y}_{i-1}$ is obtained by performing an argmax over the output softmax layer.

Hence, in addition to the intermediate hidden states and final softmax scores, the previous model prediction, $\hat{y}_{i-1}$, itself depends on the model parameters, θ, and ideally, should be backpropagated through, unlike the gold target symbol y_{i-1}^* which is independent of model parameters. However, the argmax operation is discontinuous, and thus the training objective (depicted in Figure 2.1 as the red line) exhibits discontinuities at parameter settings where the previous decoding decisions change value (depicted as changes from 'kitten' to 'dog' to 'cat'). Because these change points represent discontinuities, their gradients are undefined and the effect of correcting an earlier mistake (for example 'dog' to 'cat') as the training procedure approaches such a point is essentially hidden. In our approach, described in detail in the next section, we attempt to fix this problem by incorporating a continuous relaxation of the argmax operation into the scheduled sampling procedure in order to form an approximate but fully continuous objective. Our relaxed approximate objective is depicted in Figure 2.1 as blue and purple lines, depending on temperature parameter α which trades-off smoothness and quality of approximation.

2.3.2 Credit Assignment via Relaxation

In this section we explain in detail the continuous relaxation of greedy decoding that we will use to build a fully continuous training objective. We also introduce a related approach for sample-based training.

[1]For the sake of simplicity, the 'always sample' variant of scheduled sampling is described (Bengio et al., 2015).

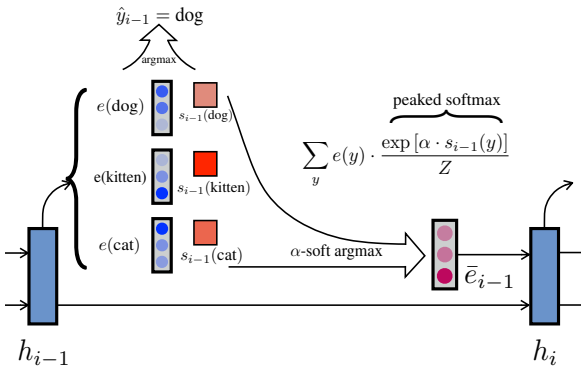

Figure 2.2: Relaxed greedy decoder that uses a continuous approximation of argmax as input to the decoder state at next time step.

Soft Argmax

In scheduled sampling, the embedding for the best scoring word at the previous step is passed as an input to the current step. This operation[2] can be expressed as

$$\hat{e}_{i-1} = \sum_y e(y)\mathbf{1}[\forall y' \neq y \;\; s_{i-1}(y) > s_{i-1}(y')]$$

where y is a word in the vocabulary, $s_{i-1}(y)$ is the output score of that word at the previous step, and $\hat{e}_{i-1}$ is the embedding passed to the next step. This operation can be relaxed by replacing the indicator function with a *peaked softmax function* with hyperparameter α to define a soft argmax procedure:

$$\bar{e}_{i-1} = \sum_y e(y) \cdot \frac{\exp(\alpha \; s_{i-1}(y))}{\sum_{y'} \exp(\alpha \; s_{i-1}(y'))}$$

As $\alpha \to \infty$, the equation above approaches the true argmax embedding. Hence, with a finite and large α, we get a linear combination of all the words (and therefore a continuous function of the parameters) that is dominated heavily by the word with maximum score.

Soft Reparametrized Sampling

Another variant of scheduled sampling is to pass a sampled embedding from the softmax distribution at the previous step to the current step instead of the argmax. This is expected to enable better exploration of the search space during optimization due to the added randomness and hence result in a more robust model. In this section, we discuss and review an approximation to the Gumbel reparametrization trick that we use as a module in our sample-based decoder. This approximation was proposed by Maddison et al. (2017) and Jang et al. (2016), who showed that the same soft argmax operation introduced above can be used for reducing variance of stochastic gradients when

[2]Assuming there are no ties for the sake of simplicity.

sampling from softmax distributions. Unlike soft argmax, this approach is not a fully continuous approximation to the sampling operation, but it does result in much more informative gradients compared to naive scheduled sampling procedure.

The Gumbel reparametrization trick shows that sampling from a categorical distribution can be refactored into sampling from a simple distribution followed by a deterministic transformation as follows: (i) sampling an independent Gumbel noise G for each element in the categorical distribution, typically done by transforming a sample from the uniform distribution: $U \sim Uniform(0, 1)$ as z, then (ii) adding it componentwise to the unnormalized score of each element, and finally (iii) taking an argmax over the vector. Using the same argmax softening procedure as above, they arrive at an approximation to the reparametrization trick which mitigates some of the gradient's variance introduced by sampling. The approximation is[3]:

$$\tilde{e}_{i-1} = \sum_y e(y) \cdot \frac{\exp\left(\alpha\left(s_{i-1}(y) + G_y\right)\right)}{\sum_{y'} \exp\left(\alpha\left(s_{i-1}(y') + G_{y'}\right)\right)}$$

We will use this 'concrete' approximation of softmax sampling in our relaxation of scheduled sampling with a sample-based decoder. We discuss details in the next section. Note that our original motivation based on removing discontinuity does not strictly apply to this sampling procedure, which still yields a stochastic gradient due to sampling from the Gumbel distribution. However, this approach is conceptually related to greedy relaxations since, here, the soft argmax reparametrization reduces gradient variance which may yield a more informative training signal. Intuitively, this approach results in the gradient of the loss to be more aware of the sampling procedure compared to naive scheduled sampling and hence carries forward information about decisions made at previous steps. The empirical results, discussed later, show similar gains to the greedy scenario.

Differentiable Relaxed Decoders

With the argmax relaxation introduced above, we have a recipe for a fully differentiable greedy decoder designed to produce informative gradients near change points. Our final training network for scheduled sampling with relaxed greedy decoding is shown in Figure 2.2. Instead of conditioning the current hidden state, h_i, on the argmax embedding from the previous step, $\hat{e}_{i-1}$, we use the α-soft argmax embedding, $\bar{e}_{i-1}$, defined in Section 2.3.2. This removes the discontinuity in the original greedy scheduled sampling objective by passing a linear combination of embeddings, dominated by the argmax, to the next step. Figure 2.1 illustrates the effect of varying α. As α increases, we more closely approximate the greedy decoder.

As in standard scheduled sampling, here we minimize the cross-entropy based loss at each time step. Hence the computational complexity of our approach is comparable to standard seq2seq training. As we discuss in Section 2.3.3, mixing model predictions randomly with ground truth symbols during training (Bengio et al., 2015; Daumé et al., 2009; Ross et al., 2011), while annealing the probability of using the ground truth with each epoch, results in better models and more stable training. As a result, training is reliant on the *annealing schedule* of two important hyperparameters:

[3]This is different from using the expected softmax embedding because our approach approximates the actual sampling process instead of linearly weighting the embeddings by their softmax probabilities

i) ground truth mixing probability and ii) the α *parameter* used for approximating the argmax function. For output prediction, at each time step, we can still output the hard argmax, depicted in Figure 2.2.

For the case of scheduled sampling with sample-based training–where decisions are sampled rather than chosen greedily (Bengio et al., 2015)–we conduct experiments using a related training procedure. Instead of using soft argmax, we use the soft sample embedding, $\tilde{e}_{i-1}$, defined in Section 2.3.2. Apart from this difference, training is carried out using the same procedure.

2.3.3 Experimental Setup

We perform experiments with machine translation (MT) and named entity recognition (NER).

Data

For MT, we use the same dataset (the German-English portion of the IWSLT 2014 machine translation evaluation campaign (Cettolo et al., 2014)), preprocessing and data splits as Ranzato et al. (2016). For named entity recognition, we use the CONLL 2003 shared task data (Tjong Kim Sang and De Meulder, 2003) for German language and use the provided data splits. We perform no preprocessing on the data.The output vocabulary length for MT is 32000 and 10 for NER.

Implementation details

For MT, we use a seq2seq model with a simple attention mechanism (Bahdanau et al., 2015), a bidirectional LSTM encoder (1 layer, 256 units), and an LSTM decoder (1 layer, 256 units). For NER, we use a seq2seq model with an LSTM encoder (1 layer, 64 units) and an LSTM decoder (1 layer, 64 units) with a fixed attention mechanism that deterministically attends to the ith input token when decoding the ith output, and hence does not involve learning of attention parameters. [4]

Hyperparameter tuning

We start by training with actual ground truth sequences for the first epoch and decay the probability of selecting the ground truth token as an inverse sigmoid (Bengio et al., 2015) of epochs with a decay strength parameter k. We also tuned for different values of α and explore the effect of varying α exponentially (annealing) with the epochs. In table 2.4.4, we report results for the best performing configuration of decay parameter and the α parameter on the validation set. To account for variance across randomly started runs, we ran multiple random restarts (RR) for all the systems evaluated

[4] *Fixed attention* refers to the scenario when we use the bidirectional LSTM encoder representation of the source sequence token at time step t while decoding at time step t instead of using a linear combination of all the input sequences weighted according to the attention parameters in the standard attention mechanism based models.

and always used the RR with the best validation set score to calculate test performance.

Comparison

We report validation and test metrics for NER and MT tasks in Table 2.4.4, F1 and BLEU respectively. 'Greedy' in the table refers to scheduled sampling with soft argmax decisions (either soft or hard) and 'Sample' refers the corresponding reparametrized sample-based decoding scenario. We compare our approach with two baselines: standard cross-entropy loss minimization for seq2seq models ('Baseline CE') and the standard scheduled sampling procedure (Bengio et al. (2015)). We report results for two variants of our approach: one with a fixed α parameter throughout the training procedure (α-soft fixed), and the other in which we vary α exponentially with the number of epochs (α-soft annealed).

2.3.4 Results

Table 2.1: Result on NER and MT. We compare our approach (α-soft argmax with fixed and annealed temperature) with standard cross entropy training (Baseline CE) and discontinuous scheduled sampling (Bengio et al. (2015)). 'Greedy' and 'Sample' refer to Section 2.3.2 and Section 2.3.2.

Training procedure	NER (F1)				MT (BLEU)			
	Dev		Test		Dev		Test	
Baseline CE	49.43		53.32		20.35		19.11	
	Greedy		Sample		Greedy		Sample	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Bengio et al. (2015)	49.75	54.83	50.90	54.60	20.52	19.85	20.40	19.69
α-soft fixed	51.65	55.88	51.13	56.25	21.32	20.28	20.48	19.69
α-soft annealed	51.43	**56.33**	50.99	54.20	21.28	20.18	21.36	**20.60**

All three approaches improve over the standard cross-entropy based seq2seq training. Moreover, both approaches using continuous relaxations (greedy and sample-based) outperform standard scheduled sampling (Bengio et al., 2015). The best results for NER were obtained with the relaxed greedy decoder with annealed α which yielded an F1 gain of +3.1 over the standard seq2seq baseline and a gain of +1.5 F1 over standard scheduled sampling. For MT, we obtain the best results with the relaxed sample-based decoder, which yielded a gain of +1.5 BLEU over standard seq2seq and a gain of +0.75 BLEU over standard scheduled sampling.

We observe that the reparametrized sample-based method, although not fully continuous end-to-end unlike the soft greedy approach, results in good performance on both the tasks, particularly MT. This might be an effect of stochastic exploration of the search space over the output sequences during training and hence we expect MT to benefit from sampling due to a much larger search

space associated with it. We also observe that annealing α results in good performance which

k	100	10	1	*Always*
NER (F1)	56.33	55.88	55.30	54.83

Table 2.2: Effect of different schedules for scheduled sampling on NER. k is the decay strength parameter. Higher k corresponds to gentler decay schedules. *Always* refers to the case when predictions at the previous predictions are always passed on as inputs to the next step.

suggests that a smoother approximation to the loss function in the initial stages of training is helpful in guiding the learning in the right direction. However, in our experiments we noticed that the performance while annealing α was sensitive to the hyperparameter associated with the annealing schedule of the mixing probability in scheduled sampling during training.

The computational complexity of our approach is comparable to that of standard seq2seq training. However, instead of a vocabulary-sized max and lookup, our approach requires a matrix multiplication. Practically, we observed that on GPU hardware, all the models for both the tasks had similar speeds which suggests that our approach leads to accuracy gains without compromising run-time. Moreover, as shown in Table 2.2, we observe that a gradual decay of mixing probability consistently compared favorably to more aggressive decay schedules. We also observed that the 'always sample' case of relaxed greedy decoding, in which we never mix in ground truth inputs (see Bengio et al. (2015)), worked well for NER but resulted in unstable training for MT. We reckon that this is an effect of large difference between the search space associated with NER and MT.

2.3.5 Conclusion

Our positive results indicate that mechanisms for credit assignment can be useful when added to the models that aim to ameliorate exposure bias. Further, our results suggest that continuous relaxations of the argmax operation can be used as effective approximations to hard decoding during training.

2.4 Beam Search-aware Training

In the previous section on differentiable scheduled sampling, techniques to ameliorate exposure bias when the inference during decoding was either greedy step-wise or involved sampling tokens at each step during decoding. However, *Beam search* is a desirable choice of test-time decoding algorithm for neural sequence models because it potentially avoids search errors made by simpler greedy methods. When *beam search* is used for decoding, the problems associated with exposure bias are likely to amplify because the teacher-forcing based training is performed in a manner that is drastically different from beam search. As a result, for cross-entropy trained models with teacher forcing, beam decoding has been observed to yield reduced test performance when compared with greedy decoding (Koehn and Knowles, 2017; Neubig, 2017; Cho et al., 2014). Therefore, in order to train models that can more effectively make use of beam search, we introduce, in this section, a new training procedure that focuses on the final loss metric (e.g. Hamming loss) evaluated on the output of beam search. While well-defined and a valid training criterion, this "direct loss" objective

is discontinuous and thus difficult to optimize. Hence, in our approach, we form a sub-differentiable surrogate objective by introducing a novel continuous approximation of the beam search decoding procedure.

2.4.1 Beam Search

The beam search procedure for obtaining the k-best sequences from a recurrent neural decoder is described in Algorithm 1. The variables are subscripted by decoding steps and the beam element. h refers to the hidden state of the recurrent decoder, f refers to the local score producing function involving recurrent network at each step. Each beam element is associated with a recurrent decoder. Therefore, a beam of size k will have k recurrent decoders. At each step, each element in the beam yields scores for candidate successors. The top k-scoring successors are chosen and their parent beam elements are stored as backpointers. The chosen successors further the recurrent network dynamics at each beam element. Once the search is run over maximum steps, the backpointers are followed from step T to step 1 to yield a sequence.

Algorithm 1 Standard Beam Search

Initialize:
  $h_{0,i} \leftarrow \vec{0}$, $e_{0,i} \leftarrow$ *embedding(¡s¿)*, $s_{0,i} \leftarrow 0$, $i = 1, \ldots, k$
for t = 0 to T **do**
  for i = 1 to k **do**
    for all $v \in V$ **do**
      $\tilde{s}_t[i, v] \leftarrow s_{t,i} + f(h_{t,i}, v)$ ▷ f is the local output scoring function
  $s_{t+1} \leftarrow$ *top-k-max*$(\tilde{s}_t)$ ▷ Top k values of the input matrix
  $b_{t+1,*}, y_{t,*} \leftarrow$ *top-k-argmax*$(\tilde{s}_t)$ ▷ Top k argmax index pairs of the input matrix
  for i = 1 to k **do**
    $e_{t+1,i} \leftarrow$ *embedding*$(y_{t,i})$
    $h_{t+1,i} \leftarrow r(h_{t,i}, e_{t+1,i})$ ▷ r is a nonlinear recurrent function that returns state at next step
$\hat{y} \leftarrow$ *follow-backpointer*$((b_{1,*}, y_{1,*}), \ldots, (b_{T,*}, y_{T,*}))$
$s(\hat{y}) \leftarrow$ *max*(s_T)

2.4.2 Objective

We denote the seq2seq model parameterized by θ as $\mathcal{M}(\theta)$. We denote the result of beam search over $\mathcal{M}(\theta)$ as $\hat{y} = \textit{Beam}(x, \mathcal{M}(\theta))$. Ideally, we would like to directly minimize a final evaluation loss, $L(\hat{y}, y^*)$, evaluated on the result of running beam search with input x and model $\mathcal{M}(\theta)$. For tractability, we assume that the evaluation loss decomposes over time steps t as: $L(\hat{y}, y^*) = \sum_{t=1}^{T} d(\hat{y}_t, y^*)$[5]. We refer to this idealized training objective that directly evaluates prediction loss

[5]This assumption does not hold for some popular evaluation metrics (e.g. BLEU). In these cases, surrogate evaluation losses such as Hamming distance can be used .

Algorithm 2 continuous-top-k-argmax

Inputs:
$s \in \mathbb{R}^{k \times |V|}$
Outputs:
$p_i \in \mathbb{R}^{k \times |V|}$, s.t. $\sum_j p_{ij} = 1, i = 1, \ldots, k$
$m \in \mathbb{R}^k = \textit{top-k-max}(s)$
for i = 1 to k **do** ▷ *peaked-softmax* will be dominated by scores closer to m_i
    $p_i = \textit{peaked-softmax}_\alpha(-(s - m_i \cdot \mathbf{1})^2)$ ▷ The square operation is element-wise

as the "direct loss" objective and define it as:

$$\min_\theta G_{\text{DL}}(x, \theta, y^*) = \min_\theta L(\textit{Beam}(x, \mathcal{M}(\theta)), y^*) \tag{2.3}$$

Unfortunately, optimizing this objective using gradient methods is difficult because the objective is discontinuous. The two sources of discontinuity are:

1. Beam search decoding (referred to as the function *Beam*) involves discrete argmax decisions and thus represents a discontinuous function.
2. The output, $\hat{y}$, of the *Beam* function, which is the input to the loss function, $L(\hat{y}, y^*)$, is discrete and hence the evaluation of the final loss is also discontinuous.

We introduce a surrogate training objective that avoids these problems and as a result is fully continuous. In order to accomplish this, we propose a continuous relaxation to the *composition* of our final loss metric, L, and our decoder function, $Beam$:

$$\textit{softLB}(x, \mathcal{M}(\theta), y^*) \approx (L \circ Beam)(x, \mathcal{M}(\theta), y^*)$$

Specifically, we form a continuous function *softLB* that seeks to approximate the result of running our decoder on input x and then evaluating the result against y^* using L. By introducing this new module, we are now able to construct our surrogate training objective:

$$\min_\theta \tilde{G}_{\text{DL}}(x, \theta, y^*) = \min_\theta \textit{softLB}(x, \mathcal{M}(\theta), y^*) \tag{2.4}$$

Specified in more detail in Section 2.4.2, our surrogate objective in Equation 2.4 will additionally take a hyperparameter α that trades approximation quality for smoothness of the objective. We first describe the standard discontinuous beam search procedure and then our training approach involving a continuous relaxation of beam search.

Discontinuity in Beam Search

Formally, *beam search* is a procedure with hyperparameter k that maintains a beam of k elements at each time step and expands each of the k elements to find the k-best candidates for the next time step. The procedure finds an approximate argmax of a scoring function defined on output sequences.

We describe beam search in the context of seq2seq models in Algorithm 1 – more specifically, for an encoder-decoder (Sutskever et al., 2014) model with a nonlinear auto-regressive decoder

(e.g. an LSTM (Hochreiter and Schmidhuber, 1997)). We define the global model score of a sequence y with length T to be the sum of local output scores at each time step of the seq2seq model: $s(y) = \sum_{t=1}^{T} f(h_t, y_t)$. In neural models, the function f is implemented as a differentiable mapping, $\mathbb{R}^{|h|} \rightarrow \mathbb{R}^{|V|}$, which yields scores for vocabulary elements using the recurrent hidden states at corresponding time steps. In our notation, $h_{t,i}$ is the hidden state of the decoder at time step t for beam element i, $e_{t,i}$ is the embedding of the output symbol at time-step t for beam element i, and $s_{t,i}$ is the cumulative model score at step t for beam element i. In Algorithm 1, we denote by $\tilde{s}_t \in \mathbb{R}^{k \times |V|}$ the cumulative candidate score matrix which represents the model score of each successor candidate in the vocabulary for each beam element. This score is obtained by adding the local output score (computed as $f(h_{t,i}, w)$) to the running total of the score for the candidate. The function r in Algorithms 1 and 3 yields successive hidden states in recurrent neural models like RNNs, LSTMs etc. The *embedding* operation maps a word in the vocabulary V, to a continuous embedding vector. Finally, backpointers at each time step to the beam elements at the previous time step are also stored for identifying the best sequence $\hat{y}$, at the conclusion of the search procedure. A backpointer at time step t for a beam element i is denoted by $b_{t,i} \in \{1, \ldots, k\}$ which points to one of the k elements at the previous beam. We denote a vector of backpointers for all the beam elements by $b_{t,*}$. The *follow-backpointer* operation takes as input backpointers ($b_{t,*}$) and candidates ($y_{t,*} \in \{1, \ldots, |V|\}^k$) for all the beam elements at each time step and traverses the sequence in reverse (from time-step T through 1) following backpointers at each time step and identifying candidate words associated with each backpointer that results in a sequence $\hat{y}$, of length T.

The procedure described in Algorithm 1 is discontinuous because of the *top-k-argmax* procedure that returns a pair of vectors corresponding to the k highest-scoring indices for backpointers and vocabulary items from the score matrix $\tilde{s}_t$. This index selection results in hard backpointers at each time step which restrict the gradient flow during backpropagation. In the next section, we describe a continuous relaxation to the *top-k-argmax* procedure which forms the crux of our approach.

Continuous Approximation to *top-k-argmax*

The key property that we use in our approximation is that for a real valued vector z, the argmax with respect to a vector of scores, s, can be approximated by a temperature controlled softmax operation. The argmax operation can be represented as:

$$\hat{z} = \sum_i z_i \mathbb{1}[\forall i' \neq i, \;\; s_i > s_{i'}],$$

which can be relaxed by replacing the indicator function with a *peaked-softmax* operation with hyperparameter α:

$$\begin{aligned} \tilde{z} = \sum_i z_i \, \frac{\exp(\alpha \, s_i)}{\sum_{i'} \exp(\alpha \, s_{i'})} &= z^T \cdot \frac{\text{elem-exp}(\alpha \, s)}{\sum_{i'} \exp(\alpha \, s_{i'})} \\ &= z^T \cdot \textit{peaked-softmax}_\alpha(\mathbf{s}) \end{aligned}$$

As $\alpha \rightarrow \infty$, $\tilde{z} \rightarrow \hat{z}$ so long as there is only one maximum value in the vector z. This *peaked-softmax* operation has been shown to be effective in recent work (Maddison et al., 2017; Jang et al., 2016;

Algorithm 3 Continuous relaxation to beam search

Initialize:
$h_{0,i} \leftarrow \vec{0}$, $e_{0,i} \leftarrow$ *embedding(¡s¿)*, $s_{0,i} \leftarrow 0$, $D_t \in \mathbb{R}^k \leftarrow \vec{0}$, $i = 1, \ldots, k$
for t = 0 to T **do**
  for all $w \in V$ **do**
    for i=1 to k **do**
      $\tilde{s}_t[i, w] \leftarrow s_{t,i} + f(h_{t,i}, w)$ ▷ f is a local output scoring function
    $\tilde{D}_{t,w} = d(w)$ ▷ $\tilde{D}_t$ is used to compute D_{t+1}
  $p_1, \ldots, p_k \leftarrow$ *continuous-top-k-argmax*$(\tilde{s}_t)$ ▷ Call Algorithm 2
  for i = 1 to k **do**
    $\tilde{b}_{t,i} \leftarrow$ *row_sum*(p_i) ▷ Soft back pointer computation
    $a_i \in \mathcal{R}^{|V|} \leftarrow$ *column_sum*(p_i) ▷ Contribution from vocabulary items
    $e_{t+1,i} \leftarrow a_i^T \times E$ ▷ Peaked distribution over the candidates to compute e, D, S
    $D_{t+1,i} \leftarrow a_i^T \cdot \tilde{D}_t$
    $s_{t+1,i} =$ *sum*$(\tilde{s}_t \odot p_i)$
    $\tilde{h}_{t,i} \leftarrow \vec{0}$
    for j = 1 to k **do** ▷ Get contributions from soft backpointers for each beam element
      $\tilde{h}_{t,i} += h_{t,j} * \tilde{b}_{t,i}[j]$
      $D_{t+1,i} += D_{t,j} * \tilde{b}_{t,i}[j]$
    $h_{t+1,i} \leftarrow r(\tilde{h}_{t,i}, e_{t+1,i})$ ▷ r is a nonlinear recurrent function that returns state at next step
$L =$ *peaked-softmax*$_\alpha(s_T) \cdot D_T$ ▷ Pick the loss for the sequence with highest model score on the beam in a soft manner.

Goyal et al., 2017b) involving continuous relaxation to the argmax operation, although to our knowledge, this is the first work to apply it to approximate the beam search procedure.

Using this *peaked-softmax* operation, we propose an iterative algorithm for computing a continuous relaxation to the *top-k-argmax* procedure in Algorithm 2 which takes as input a score matrix of size $k \times |V|$ and returns k peaked matrices p of size $k \times |V|$. Each matrix p_i represents the index of i-th max. For example, p_1 will have most of its mass concentrated on the index in the matrix that corresponds to the argmax, while p_2 will have most of its mass concentrated on the index of the 2nd-highest scoring element. Specifically, we obtain matrix p_i by computing the squared difference between the i-highest score and all the scores in the matrix and then using the *peaked-softmax* operation over the negative squared differences. This results in scores closer to the i-highest score to have a higher mass than scores far away from the i-highest score.

Hence, the continuous relaxation to *top-k-argmax* operation can be simply implemented by iteratively using the *max* operation which is continuous and allows for gradient flow during back-propagation. As $\alpha \to \infty$, each p vector converges to hard index pairs representing hard backpointers and successor candidates described in Algorithm 1. For finite α, we introduce a notion of a soft backpointer, represented as a vector $\tilde{b} \in \mathbb{R}^k$ in the k-probability simplex, which represents the contribution of each beam element from the previous time step to a beam element at current time step. This is obtained by a row-wise sum over p to get k values representing soft backpointers.

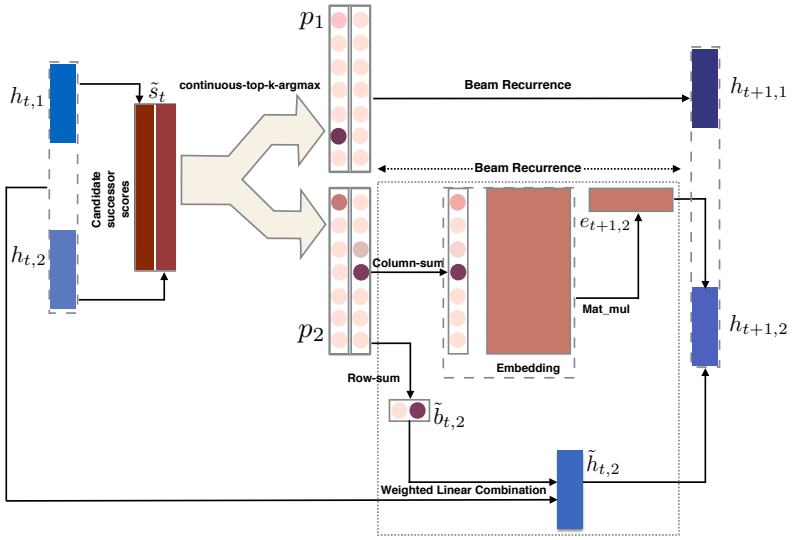

Figure 2.3: Illustration of our approximate continuous beam search (Algorithm 3) module to obtain hidden states for beam elements at the next time step ($h_{t+1,*}$), starting from the hidden states corresponding to beam elements are current time step ($h_{t,*}$) with beam size of 2. 'Beam recurrence' module has been expanded for $h_{t+1,2}$ and similar procedure is carried out for $h_{t+1,1}$.

Training with Continuous Relaxation of Beam Search

We describe our approach in detail in Algorithm 3 and illustrate the soft beam recurrence step in Figure 1. For composing the loss function and the beam search function for our optimization as proposed in Equation 2, we make use of decomposability of the loss function across time-steps. Thus for a sequence y, the total loss is: $L(y, y^*) = \sum_{t=1}^{T} d(y_t)$. In our experiments, $d(y_t)$ is the Hamming loss which can be easily computed at each time-step by simply comparing gold y_t^* with y_t. While exact computation of $d(y)$ will vary according to the loss, our proposed procedure will be applicable as long as the total loss is decomposable across time-steps. While decomposability of loss is a strong assumption, existing literature on structured prediction (Taskar et al., 2004; Tsochantaridis et al., 2005) has made due with this assumption, often using decomposable losses as surrogates for non-decomposable ones. We detail the continuous relaxation to beam search in Algorithm 3 with $D_{t,i}$ being the cumulative loss of beam element i at time step t and E being the embedding matrix of the target vocabulary which is of size $|V| \times l$ where l is the size of the embedding vector.

In Algorithm 3, all the discrete selection functions have been replaced by their soft, continuous counterparts which can be backpropagated through. This results in all the operations being matrix and vector operations which is ideal for a GPU implementation. An important aspect of this algorithm is that we no longer rely on exactly identifying a discrete search prediction $\hat{y}$ since we are only interested in a continuous approximation to the direct loss L (line 18 of Algorithm 3),

and all the computation is expressed via the *soft beam search* formulation which eliminates all the sources of discontinuities associated with the training objective in Equation 1. The computational complexity of our approach for training scales linearly with the beam size and hence is roughly k times slower than standard CE training for beam size k. Since we have established the pointwise convergence of *peaked-softmax* to argmax as $\alpha \to \infty$ for all vectors that have a unique maximum value, we can establish pointwise convergence of objective in Equation 2 to objective in Equation 1 as $\alpha \to \infty$, as long as there are no ties among the top-k scores of the beam expansion candidates at any time step. We posit that absolute ties are unlikely due to random initialization of weights and the domain of the scores being $\mathbb{R}$. Empirically, we did not observe any noticeable impact of potential ties on the training procedure and our approach performed well on the tasks as discussed in Section 2.4.4.

$$\tilde{G}_{\text{DL},\alpha}(x, \theta, y^*) \xrightarrow[p]{\alpha \to \infty} G_{\text{DL}}(x, \theta, y^*) \tag{2.5}$$

We experimented with different annealing schedules for α starting with *non-peaked softmax* moving toward *peaked-softmax* across epochs so that learning is stable with informative gradients. This is important because cost functions like Hamming distance with very high α tend to be non-smooth and are generally flat in regions far away from changepoints and have a very large gradient near the changepoints which makes optimization difficult.

Decoding

The motivation behind our approach is to make the optimization aware of beam search decoding while maintaining the continuity of the objective. However, since our approach doesn't introduce any new model parameters and optimization is agnostic to the architecture of the seq2seq model, we were able to experiment with various decoding schemes like locally normalized greedy decoding, and hard beam search, once the model has been trained.

However, to reduce the gap between the training procedure and test procedure, we also experimented with *soft beam search decoding*. This decoding approach closely follows Algorithm 3, but along with soft back pointers, we also compute hard back pointers at each time step. After computing all the relevant quantities like model score, loss etc., we follow the *hard* backpointers to obtain the best sequence $\hat{y}$. This is very different from hard beam decoding because at each time step, the selection decisions are made via our soft continuous relaxation which influences the scores, LSTM hidden states and input embeddings at subsequent time-steps. The hard backpointers are essentially the MAP estimate of the soft backpointers at each step. With small, finite α, we observe differences between soft beam search and hard beam search decoding in our experiments.

Comparison with Max-Margin Objectives

Max-margin based objectives are typically motivated as another kind of surrogate training objective which avoid the discontinuities associated with direct loss optimization. Hinge loss for structured prediction typically takes the form:

$$G_{\textit{hinge}} = \max(0, \max_{y \in \mathcal{Y}}(\Delta(y, y^*) + s(y)) - s(y^*))$$

where x is the input sequence, y^* is the gold target sequence, $\mathcal{Y}$ is the output search space and $\Delta(y, y^*)$ is the discontinuous cost function which we assume is decomposable across the time-steps of a sequence. Finding the cost augmented maximum score is generally difficult in large structured models and often involves searching over the output space and computing the approximate cost augmented maximal output sequence and the score associated with it via beam search. This procedure introduces discontinuities in the training procedure of structured max-margin objectives and renders it non amenable to training via backpropagation. Related work (Wiseman and Rush, 2016) on incorporating beam search into the training of neural sequence models does involve cost-augmented max-margin loss but it relies on discontinuous beam search forward passes and an explicit mechanism to ensure that the gold sequence stays in the beam during training, and hence does not involve back propagation through the beam search procedure itself.

Our continuous approximation to beam search can very easily be modified to compute an approximation to the structured hinge loss so that it can be trained via backpropagation if the cost function is decomposable across time-steps. In Algorithm 3, we only need to modify line 5 as:

$$\tilde{s}_t[i, w] \leftarrow s_{t,i} + d(w) + f(h_{t,i}, w)$$

and instead of computing L in Algorithm 3, we first compute the cost augmented maximum score as:

$$s_{max} = \textit{peaked-softmax}_{\alpha}(s_T) \cdot s_T$$

and also compute the target score $s(y^*)$ by simply running the forward pass of the LSTM decoder over the gold target sequence. The continuous approximation to the hinge loss to be optimized is then: $\tilde{G}_{\text{hinge},\alpha} = \max(0, s_{max} - s(y^*))$. We empirically compare this approach with the proposed approach to optimize direct loss in experiments.

2.4.3 Experimental Setup

Since our goal is to investigate the efficacy of our approach for training generic seq2seq models, we perform experiments on two NLP tagging tasks with very different characteristics and output search spaces: Named Entity Recognition (NER) and CCG supertagging. While seq2seq models are appropriate for CCG supertagging task because of the long-range correlations between the sequential output elements and a large search space, they are not ideal for NER which has a considerably smaller search space and weaker correlations between predictions at subsequent time steps. In our experiments, we observe improvements from our approach on *both* of the tasks. We use a seq2seq model with a bi-directional LSTM encoder (1 layer with tanh activation function) for the input sequence x, and an LSTM decoder (1 layer with tanh activation function) with a fixed attention mechanism that deterministically attends to the i-th input token when decoding the i-th output, and hence does not involve learning of any attention parameters. Since, computational complexity of our approach for optimization scales linearly with beam size for each instance, it is impractical to use very large beam sizes for training. Hence, beam size for all the beam search based experiments was set to 3 which resulted in improvements on both the tasks as discussed in the results. For both tasks, the direct loss function was the *Hamming distance cost* which aims to maximize word level accuracy.

Named Entity Recognition

For named entity recognition, we use the CONLL 2003 shared task data (Tjong Kim Sang and De Meulder, 2003) for German language and use the provided data splits. We perform no preprocessing on the data. The output vocabulary size (label space) is 10. A peculiar characteristic of this problem is that the training data is naturally skewed toward one default label ('O') because sentences typically do not contain many named entities and the evaluation focuses on the performance recognizing entities. Therefore, we modify the Hamming cost such that incorrect prediction of 'O' is doubly penalized compared to other incorrect predictions. We use the hidden layers of size 64 and label embeddings of size 8. As mentioned earlier, seq2seq models are not an ideal choice for NER (tag-level correlations are short-ranged in NER – the unnecessary expressivity of full seq2seq models over simple encoder-classifier neural models makes training harder). However, we wanted to evaluate the effectiveness of our approach on different instantiations of seq2seq models.

CCG Supertagging

We used the standard splits of CCG bank (Hockenmaier and Steedman, 2002) for training, development, and testing. The label space of supertags is 1,284 which is much larger than NER. The distribution of supertags in the training data exhibits a long tail because these supertags encode specific syntactic information about the words' usage. The supertag labels are correlated with each other and many tags encode similar information about the syntax. Moreover, this task is sensitive to the long range sequential decisions and search effects because of how it holistically encodes the syntax of the entire sentence. We perform minor preprocessing on the data similar to the preprocessing in Vaswani et al. (2016). For this task, we used hidden layers of size 512 and the supertag label embeddings were also of size 512. The standard evaluation metric for this task is the word level label accuracy which directly corresponds to Hamming loss.

Hyperparameter tuning

For tuning all the hyperparameters related to optimization we trained our models for 50 epochs and picked the models with the best performance on the development set. We also ran multiple random restarts for all the systems evaluated to account for performance variance across randomly started runs. We pretrained all our models with standard cross entropy training which was important for stable optimization of the non convex neural objective with a large parameter search space. This warm starting is a common practice in prior work on complex neural models (Ranzato et al., 2016; Rush et al., 2015; Bengio et al., 2015).

Comparison

We report performance on validation and test sets for both the tasks in Tables 1 and 2. The baseline model is a cross entropy trained seq2seq model (Baseline CE) which is also used to warm start the the proposed optimization procedures in this paper. This baseline has been compared against the approximate direct loss training objective (Section 2.4.2), referred to as $\tilde{G}_{\text{DL},\alpha}$ in the tables, and the approximate max-margin training objective (Section 2.4.2), referred to as $\tilde{G}_{\text{hinge},\alpha}$ in the tables.

Training procedure	Greedy		Hard Beam Search		Soft Beam Search	
	Dev	Test	Dev	Test	Dev	Test
Baseline CE	80.15	80.35	82.17	82.42	81.62	82.00
$\tilde{G}_{\text{hinge},\alpha}$ annealed α	-	-	83.03	83.54	82.82	83.05
$\tilde{G}_{\text{hinge},\alpha}\alpha$=1.0	-	-	83.02	83.36	82.49	82.85
$\tilde{G}_{\text{DL},\alpha}\alpha$=1.0	-	-	83.23	82.65	82.58	82.82
$\tilde{G}_{\text{DL},\alpha}$ annealed α	-	-	**85.69**	**85.82**	85.58	85.78

Table 2.3: Results on CCG Supertagging. Tag-level accuracy is reported in this table which is a standard evaluation metric for supertagging.

Training procedure	CE Greedy		Hard Beam Search		Soft Beam Search	
	Dev	Test	Dev	Test	Dev	Test
Baseline CE	50.21	54.92	46.22	51.34	47.50	52.78
$\tilde{G}_{\text{hinge},\alpha}$ annealed α	-	-	41.10	45.98	41.24	46.34
$\tilde{G}_{\text{hinge},\alpha}\alpha$=1.0	-	-	40.09	44.67	39.67	43.82
$\tilde{G}_{\text{DL},\alpha}\alpha$=1.0	-	-	49.88	54.08	50.73	54.77
$\tilde{G}_{\text{DL},\alpha}$ annealed α	-	-	51.86	56.15	**51.96**	**56.38**

Table 2.4: Results on Named Entity Recognition. Macro F1 over the prediction of different named entities is reported that is a standard evaluation metric for this task.

Results are reported for models when trained with annealing α, and also with a constant setting of $\alpha = 1.0$ which is a very smooth but inaccurate approximation of the original direct loss that we aim to optimize[6]. Comparisons have been made on the basis of performance of the models under different decoding paradigms (represented as different column in the tables): locally normalized decoding (CE greedy), hard beam search decoding and soft beam search decoding described in Section 2.4.2.

2.4.4 Results

As shown in Tables 1 and 2, our approach $\tilde{G}_{\text{DL},\alpha}$ shows significant improvements over the locally normalized CE baseline with greedy decoding for both the tasks (+5.5 accuracy points gain for supertagging and +1.5 F1 points for NER). The improvement is more pronounced on the supertagging task, which is not surprising because: (i) the evaluation metric is tag-level accuracy which is congruent with the Hamming loss that $\tilde{G}_{\text{DL},\alpha}$ directly optimizes and (ii) the supertagging task itself is very sensitive to the search procedure because tags across time-steps tend to exhibit long range dependencies as they encode specialized syntactic information about word usage in the sentence.

Another common trend to observe is that annealing α always results in better performance than training with a constant $\alpha = 1.0$ for both $\tilde{G}_{\text{DL},\alpha}$ (Section 2.4.2) and $\tilde{G}_{\text{hinge},\alpha}$ (Section 2.4.2).

[6]Our pilot experiments that involved training with a very large constant α resulted in unstable optimization.

This shows that a stable training scheme that smoothly approaches minimizing the actual direct loss is important for our proposed approach. Additionally, we did not observe a large difference when our soft approximation is used for decoding (Section 2.4.2) compared to hard beam search decoding, which suggests that our approximation to the hard beam search is as effective as its discrete counterpart.

For supertagging, we observe that the baseline cross entropy trained model improves its predictions with beam search decoding compared to greedy decoding by 2 accuracy points, which suggests that beam search is already helpful for this task, even without search-aware training. Both the optimization schemes proposed in this paper improve upon the baseline with soft direct loss optimization ($\tilde{G}_{\text{DL},\alpha}$), performing better than the approximate max-margin approach. [7]

For NER, we observe that optimizing $\tilde{G}_{\text{DL},\alpha}$ outperforms all the other approaches but we also observe interesting behaviour of beam search decoding and the approximate max-margin objective for this task. The pretrained CE baseline model yields worse performance when beam search is done instead of greedy locally normalized decoding. This is because the training data is heavily skewed toward the 'O' label and hence the absolute score resolution between different tags at each time-step during decoding isn't enough to avoid leading beam search toward a wrong hypothesis path. We observed in our experiments that hard beam search resulted in predicting more 'O's which also hurt the prediction of tags at future time steps and hurt precision as well as recall. Encouragingly, $\tilde{G}_{\text{DL},\alpha}$ optimization, even though warm started with a CE trained model that performs worse with beam search, led to the NER model becoming more search aware, which resulted in superior performance. However, we also observe that the approximate max-margin approach ($\tilde{G}_{\text{hinge},\alpha}$) performs poorly here. We attribute this to a deficiency in the max-margin objective when coupled with approximate search methods like beam search that do not provide guarantees on finding the supremum: one way to drive this objective down is to learn model scores such that the search for the best hypothesis is difficult, so that the value of the loss augmented decode is low, while the gold sequence maintains higher model score. Because we also warm started with a pre-trained model that results in a worse performance with beam search decode than with greedy decode, we observe the adverse effect of this deficiency. The result is a model that scores the gold hypothesis highly, but yields poor decoding outputs. This observation indicates that using max-margin based objectives with beam search during *training* actually may achieve the opposite of our original intent: the objective can be driven down by *introducing* search errors.

The observation that our optimization method led to improvements on *both* the tasks–even on NER for which hard beam search during decoding on a CE trained model hurt the performance–by making the optimization more search aware, indicates the effectiveness of our approach for training seq2seq models.

[7] Separately, we also ran experiments with a max-margin objective that used hard beam search to compute loss-augmented decodes. This objective is discontinuous, but we evaluated the performance of gradient optimization nonetheless. While not included in the result tables, we found that this approach was unstable and considerably underperformed both approximate max-margin and direct loss objectives.

2.4.5 Conclusion and Proposed extension

While beam search is a method of choice for performing search in neural sequence models, as our experiments confirm, it is not necessarily guaranteed to improve accuracy when applied to cross-entropy-trained models. We proposed a novel method for optimizing model parameters that directly takes into account the process of beam search itself through a continuous, end-to-end sub-differentiable relaxation of beam search composed with the final evaluation loss. Experiments demonstrate that our method is able to improve overall test-time results for models using beam search as a test-time inference method, leading to substantial improvements in accuracy.

As discussed earlier, another closely related work to incorporating beam search into training of sequence models is work by Wiseman and Rush (2016) (BSO) which relies on discontinuous beam search forward passes and involves a variant of cost-augmented max-margin loss and an explicit loss to ensure that the gold sequence stays in the beam during training. This training scheme and loss function was found to be effective for tasks like Machine Translation and it would be interesting to study the difference between this training strategy that focuses on designing better loss functions and the training strategy described in our work that focuses on noisy gradients of the continuous relaxation to beam search. Hence, I propose to implement BSO and perform comparable experiments for my thesis.

Chapter 3

Globally Normalized Sequence Models: Ameliorating Label Bias

In this chapter, we explore the advantages that global normalization provides over local normalization. In particular, we identify the two sources of *label bias* associated with locally normalized sequence models and propose a beam-search aware training algorithm for globally normalized models that helps ameliorate label bias. Moreover, we also recognize that globally normalized models naturally enable us to compute marginal probabilities of a subset of variables in a sequence and exploit this fact to incorporate external expectation-based constraints on the marginal posterior of globally normalized models. With this posterior regularization, we perform learning of multiple tagging tasks jointly using conditional random fields.

3.1 Label Bias

Locally normalized models are often associated with *label bias* because at each step they are constrained to put the entire probabilistic mass on all possible labels at the step. *Label bias* is a phenomenon when the model relies mostly on its prediction history for predicting the next item in the sequence and largely ignores the input. This is undesirable especially in the cases when the model's prediction is not correct and heavy reliance on the model's predicted history leads to a cascade of errors in sequence prediction. As mentioned earlier, since locally normalized models are constrained to allocate the entire probability mass to all the next possible items in the vocabulary for the next time step, they are likely to lead to putting a larger amount of mass than desired on undesirable sequences. In this thesis, we identify two major source of label bias: i) conditioning on partial input, and ii) inexact inference. The first cause has been studied in great detail in the context of comparing the locally normalized maximum entropy markov models(MEMMs) with the globally normalized conditional random fields(CRFs). However, in this thesis we argue that this source of label bias is largely eliminated by expressive neural encoders that are capable of representing information about the entire input. Instead, we demonstrate empirically that locally normalized models trained using a search-aware technique still suffer from label bias largely arising due to inexact search.

3.1.1 Label Bias with partial input

It was shown in (Andor et al., 2016; Lafferty et al., 2001), locally normalized conditional models *with access to only partial input*, $x_{1:i-1}$, at each decoding step are biased towards labeling decisions with low-entropy transition probabilities at each decoding step and, as a result, suffer from a weakened ability to revise previous decisions based upon future input observations. This phenomenon has been referred to as *label bias*, and presents itself as an arbitrary allocation of probability mass to unlikely or undesirable label sequences despite the presence of well-formed sequences in training data. Andor et al. (2016) prove that this class of locally normalized models that relies on the structural assumption of access to only left-to-right partial input at each step,

$$\prod_{i=1}^{n} p(y_i \mid \mathbf{x}, y_{1:i-1}) = \prod_{i=1}^{n} p(y_i \mid x_{1:i-1}, y_{1:i-1}),$$

is strictly less expressive than its globally normalized counterpart.

However, the standard sequence-to-sequence models used most often in practice and presented in this paper actually condition the decoder on a summary representation of the *entire input sequence*, $\mathbf{x}$, computed by a neural encoder. Hence, depending on the power of the encoder, it is commonly thought that such models avoid this type of label bias. For these models, both locally normalized and globally normalized conditional models are equally expressive, in principle, with a sufficiently powerful encoder.

However, as we suggest in the next section and show empirically in experiments, this does not necessarily mean that both parametrizations are equally amenable to gradient-based training in practice, particularly when the search space is large and search-aware training techniques are used.

3.1.2 Label Bias due to approximate search

To improve performance with inexact decoding methods (e.g. beam search), search-aware training techniques take into account the decoding procedure that will be used at test time and adjust the parameters of the model to maximize prediction accuracy under the decoder. Because of the popularity of beam search as a decoding procedure for sequence models, we focus on beam search-aware training. While many options are available, including beam-search optimization (BSO) (Wiseman and Rush, 2016), we use search-aware training strategy based upon continuous relaxation to beam search described in section 2.

We illustrate via example how optimization of locally normalized models may suffer from a new kind of label bias when using beam search-aware training, and point to reasons why this issue might be mitigated by the use of globally normalized models. While the scores of successors of a single candidate under a locally normalized model are constrained to sum to one, scores of successors under a globally normalized model need only be positive. Intuitively, during training, this gives the globally normalized model more freedom to downweight undesirable intermediate candidates in order avoid search errors.

In the example beam search decoding problem in Figure 3.1, we compare the behavior of locally and globally normalized models at a single time step for a beam size of two. In this example, we

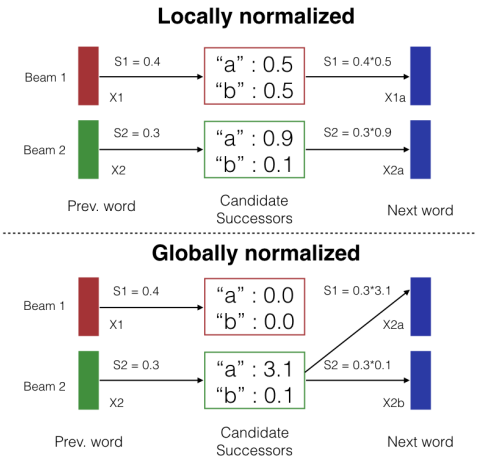

Figure 3.1: Illustrative example of bias arising in locally normalized models due to beam search. Red indicates the candidate that optimization should learn to discard and green indicates the candidate that should be propagated. Locally normalized models are constrained to return normalized scores for the successors of each candidate, while globally normalized models are unconstrained and can more easily learn to drop successors of the red candidate.

assume that the score for beams in both the models is exactly the same until the step shown in Figure 3.1. Suppose that the lower item on the beam(X2) is correct, and thus, for more effective search, we would prefer the models scores to be such that only successors of the lower beam item are present on the beam at the next step. However, since, the scores at each step for a locally normalized model are constrained to sum to one, the upper beam item(X1) generates successors with scores comparable to those of the lower beam item. As we see in the example, due to the normalization constraint, search-aware training of the locally normalized model might find it difficult to set the parameters to prevent extension of the poorer candidate. In contrast, because the scores of a *globally normalized* model are not constrained to sum to one, the parameters of the neural model can be set such that *all* the successors of the bad candidate have a very low score and thus do not compete for space on the beam. This illustrates a mechanism by which search-aware training of globally normalized models in a large search spaces might be more effective. However as discussed earlier, if we can perform *exact search* then this *label bias* ceases to exist because both the models have the same expressive power with a search-agnostic optimization scheme. In experiments, we will explore this trade-off empirically.

3.2 Search-aware Training for Globally Normalized Models

In this section, we investigate the effectiveness of globally normalized sequence models at ameliorating label bias when compared to similarly parametrized locally normalized models. We extend the previously described approach for search-aware training via a continuous relaxation of beam search (Goyal et al., 2017c) in order to enable training of *globally normalized* recurrent sequence models through simple backpropagation. (Smith and Johnson, 2007) proved that locally normalized conditional PCFGs and unnormalized conditional WCFGs are equally expressive for finite length

sequences and posit that Maximum Entropy Markov Models (MEMMs) are weaker than CRFs because of the structural assumptions involved with MEMMs that result in label bias. We find out in our experiment that in spite of having the same expressive power, training globally normalized models is easier and results in better models, when search-aware training is done in the context of huge search space which rules out exact search and non-convex optimization. For controlled comparison, we experiment with the same underlying neural architecture for training both locally and globally normalized models in a search-aware manner. We then use the extension to the beam search aware training to conduct an empirical study of the interaction between global normalization, high-capacity encoders, and search-aware optimization. In particular, a major modification is that while for training search-aware locally normalized models via continuous relaxation to beam search, we use *log-normalized* successor scores at each decode step, for training globally normalized models however, we directly use *unnormalized scores* at each time step, which are $\in \mathrm{R}_{+}$. This has an effect of associating a non-negative score with each sequence in the finite hypothesis space which has to be explicitly normalized in order to get the probability of the sequence. Since, a probabilistic formulation is not necessary for beam search to work, beam-search aware training allows us to train *unnormalized* models.

However, we note that our proposed approach is not the only approach to train globally normalized sequence models. Most other alternatives for search-aware training of globally normalized neural sequence models include approaches that use some mechanism like early updates (Collins and Roark, 2004) that relies on explicitly tracking if the gold sequence falls off the beam and is not end-to-end continuous. (Andor et al., 2016) describe a method for training globally normalized neural feedforward models, which involves optimizing a CRF-based likelihood where the normalizer is approximated by the sum of the scores of the final beam elements. They describe and focus on label bias arising out of conditioning on partial input and hence focused on the scenario in which locally normalized models can be less expressive than globally normalized models, whereas we also consider another source of label bias which might be affecting the optimization of equally expressive locally and globally normalized conditional models. (Wiseman and Rush, 2016) also propose a beam search based training procedure that uses unnormalized scores similar to our approach. Their models achieve good performance over CE baselines – a pattern that we observe in our results as well. Our approach to training unnormalized models however is end-to-end sub-differentiable and the whole training procedure can be encoded via a computation graph in an autodiff library. Our main focus is to empirically analyze the factors affecting the boost in performance with end-to-end continuous search-aware training (Goyal et al., 2017c) for globally normalized models. We observe that in the context of inexact search, globally normalized neural models are still more effective than their locally normalized counterparts.

Further, since our training approach is sensitive to warm-starting with pre-trained models, we also propose a novel initialization strategy based on self-normalization for pre-training globally normalized models.

3.2.1 Initialization for training globally normalized models

Goyal et al. (2017c) reported that *initialization* with a locally normalized model pre-trained with teacher-forcing was important for their continuous beam search based approach to be stable and

hence they used the locally normalized log-scores for their search-aware training model. In this work, we experimented with the unnormalized candidate successor scores and found that initializing the optimization for a globally normalized objective with a cross-entropy trained locally normalized model resulted in unstable training. This is expected because the locally normalized models are parametrized in a way such that using the scores before the softmax normalization results in a very different outcome than using scores after local normalization. For example, the locally normalized Machine Translation model in Table 3.1, that gives a BLEU score of 27.62 when decoded with beam search using locally normalized scores, results in BLEU of 4.30 when beam search decoding is performed with unnormalized scores. Pre-training a truly globally normalized model for initialization is not straghtforward because no exact likelihood maximization techniques exist for globally normalized models as the global normalizer is intractable to compute.

Therefore, we propose a new approach to initialization for search-aware training of globally normalized models: we pre-train a locally normalized model that is parametrized like a globally normalized model. More specifically, we train a locally normalized model with its distribution over the output sequences denoted by $p_L(\mathcal{Y})$ such that we can easily find a globally normalized model with a distribution $p_G(\mathcal{Y})$ that matches $p_L(\mathcal{Y})$. Following the notation in Section 2, for a locally normalized model, the log-probability of a sequence is:

$$\sum_{i=1}^{n} [\log s(\mathbf{x}, y_{1:i-1}, y_i) - \log Z_{L,i}(\mathbf{x}, y_{1:i-1})]$$

and for a globally normalized model it is:

$$\left[\sum_{i=1}^{n} \log s(\mathbf{x}, y_{1:i-1}, y_i)\right] - \log Z_G(\mathbf{x})$$

Self Normalization

One way to find a locally normalized model that is parametrized like a globally normalized model is to ensure that the local normalizer at each step, $\log\ Z_{L,i}(\mathbf{x}, y_{1:i-1})$, is 0. With the local normalizer being zero it is straightforward to see that the log probability of a sequence under a locally normalized model can easily be interpreted as log probability of the sequence under a globally normalized model with the global log-normalizer, $\log\ Z_G(\mathbf{x}) = 0$. This training technique is called *self-normalization* (Andreas and Klein, 2015) because the resulting models' unnormalized score at each step lies on a probability simplex. A common technique for training self-normalized models is L2-regularization of local log normalizer which encourages learning a model with $\log\ Z = 0$ and was found to be effective for learning a language model by Devlin et al. (2014)[1]. The L2-regularized

[1]Noise Contrastive Estimation (Mnih and Teh, 2012; Gutmann and Hyvärinen, 2010) is also an alternative to train unnormalized models but our experiments with NCE were unstable and resulted in worse models.

cross entropy objective is given by:

$$\min_{\theta} \sum_{\mathbf{x},\mathbf{y}^* \in \mathcal{D}} -\sum_{i=1}^{n} \log\ p(y_i^* \mid \mathbf{x}, y_{1:i-1}) + \lambda \cdot (\log\ Z_{L,i}(\mathbf{x}, y_{1:i-1}))^2$$

In Table 3.1, we report the mean and variance of the local log normalizer on the two different tasks using L2-regularization (*L2*) based self normalization and no self normalization (*CE*). We observe that *L2* models are competitive performance-wise to the cross-entropy trained locally normalized models while resulting in a much smaller local log-normalizer on average. Although, we couldn't minimize $\log\ Z$ exactly to 0, we observe in Section 4 that this is sufficient to train a reasonable initializer for the search-aware optimization of globally normalized models. It is important to note that these approaches yield a *globally normalized* model that is equivalent to a locally normalized model trained via teacher-forcing and hence these are only used to *warm-start* the search-aware optimization of globally normalized models. Our search-aware training approach is free to adjust the parameters of the models such that the final globally normalized model has a non-zero log-normalizer Z_G over the data.

		Train logZ		Dev logZ		Acc/
		Mean	Var	Mean	Var	BLEU
CCG	CE	21.08	9.57	21.96	9.18	93.3
	L2	0.6	0.29	0.26	0.08	91.9
MT	CE	24.7	115.4	25.8	129.1	27.62
	L2	0.65	0.18	0.7	0.29	26.63

Table 3.1: Comparison of logZ between cross entropy trained models (CE) and self normalized models (L2) for CCG supertagging and Machine Translation tasks.

Other possible approaches to project locally normalized models onto globally normalized models include distribution matching via knowledge distillation (Hinton et al., 2015). We leave exploration of warm-starting of search aware optimization with this approach to future work.

3.2.2 Experimental setup

To empirically analyze the interaction between label bias arising from different sources, search-aware training, and global normalization, we conducted experiments on two tasks with vastly different sizes of output space: CCG supertagging and Machine Translation. As described in the next section, the task of tagging allows us to perform controlled experiments which explicitly study the effect of amount of input information available to the decoder at each step, we analyze the scenarios in which search aware training and global normalization are expected to improve the model performance.

In all our experiments, we report results on training with standard teacher forcing optimization and self-normalization as our baselines. We report results with both search-aware locally and globally normalized models (Section 3.1) after warm starting with both cross entropy trained models

and self-normalized models to study the effects of search-aware optimization and global normalization. We follow Goyal et al. (2017c) and use the decomposable Hamming loss approximation with search-aware optimization for both the tasks and decode via *soft beam search decoding* method which involves continuous beam search with soft backpointers for the LSTM Beam search dynamics as described in Section 3, but using identifiable backpointers and labels (using MAP estimates of soft backpointers and labels) to decode.

We tune hyperparameters like learning rate and annealing schedule by observing performance on development sets for both the tasks. We performed at least three random restarts for each class and report results based on best development performance.

CCG supertagging

We used the standard splits of CCG bank (Hockenmaier and Steedman, 2002) for training, development, and testing. The label space of supertags is 1,284 and the labels are correlated with each other based on their syntactic relations. The distribution of supertag labels in the training data exhibits a long tail distribution. This task is sensitive to the long range sequential decisions because it encodes rich syntactic information about the sentence. Hence, this task is ideal to analyze the effects of label bias and search effects. We perform minor preprocessing on the data similar to the preprocessing in Vaswani et al. (2016). For experiments related to search aware optimization, we report results with beam size of 5.[2]

Tagging model for ablation study

We changed the standard sequence-to-sequence model to be more suitable for the tagging task. This change also lets us perform controlled experiments pertaining to the amount of input sequence information available to the decoder at each time step.

In a standard encoder-decoder model with attention, the initial hidden state of the decoder is often some function of the final encoder state so that the decoder's predictions can be conditioned on the full input. For our tagging experiments, instead of influencing the initial decoder state with the encoder, we set it to a vector of zeros. Thus the information about input for prediction is *only* available via the attention mechanism. In addition to the change above, we also forced the model to attend to only the i^{th} input representation while predicting the i^{th} label. This is enforceable because the output length is equal to the input length and it is also a more suitable structure for a tagging model. With these changes in the decoder, we can precisely control the amount of information about the input available to the decoder at each prediction step. For example, with a unidirectional LSTM encoder, the decoder at i^{th} step only has access to input till the i^{th} token and the prediction history:

$$p(y_i \mid \mathbf{x}, y_{1:i-1}) = p(y_i \mid x_{1:i}, y_{1:i-1})$$

This setting lets us clearly explore the classical notion of *label bias* arising out of access to partial input at each prediction step (Section 2.3). A bidirectional LSTM encoder, however provides access to all of the input information to the decoder at all the prediction steps.

[2]We observed similar results with beam size 10

	Unidirectional	Bidirectional
pretrain-greedy	76.54	92.59
pretrain-beam	77.76	93.29
locally normalized	83.9	**93.76**
globally normalized	**83.93**	93.73

Table 3.2: Accuracy results on CCG supertagging when initialized with a regular teacher-forcing model. Reported using *Unidirectional* and *Bidirectional* encoders respectively with fixed attention tagging decoder. *pretrain-greedy* and *pretrain-beam* refer to the output of decoding the initializer model. *locally normalized* and *globally normalized* refer to search-aware soft-beam models

	Unidirectional	Bidirectional
pretrain-greedy	73.12	91.23
pretrain-beam	73.83	91.94
locally normalized	83.35	**92.78**
globally normalized	**85.50**	92.63

Table 3.3: Accuracy results on CCG supertagging when initialized with a self normalized model.

Machine Translation

We use the same dataset (the German-English portion of the IWSLT 2014 machine translation evaluation campaign (Cettolo et al., 2014)), preprocessing and data splits as Ranzato et al. (2016) for our Machine Translation experiments. The output label/vocabulary size is 32000 and unlike tagging, the length of output sequences cannot be deterministically determined from the length of the input sequence. Moreover, the output sequence does not necessarily align monotonically with the input sequence. Hence the output sequence space for MT is much larger than that for tagging and the effects of inexact search on optimization are expected to be even more apparent for MT. We use a standard LSTM-based encoder/decoder model with a standard attention mechanism (Bahdanau et al., 2016) for our MT experiments. For search-aware optimization experiments, we report results with beam size 3.[3]

3.2.3 Results and Analysis

The results reported in Tables 3.2, 3.3 and 3.4 allow us to analyze the effect of interaction of label bias, inexact search and global normalization in detail.

[3]We observed similar results beam size of 5.

Init-scheme →	Regular	Self-normalized
pretrain-greedy	26.24	25.42
pretrain-beam	27.62	26.63
locally-normalized	**29.28**	27.71
globally-normalized	26.24	**29.27**

Table 3.4: BLEU results on de-en Machine Translation. *Regular* and *Self-normalized* refer to the initization scheme for soft beam search training. *pretrain-greedy* and *pretrain-beam* refer to the output of decoding the initializer model. *locally normalized* and *globally normalized* refer to search-aware soft-beam models

Label bias with partial input

First, we analyze the effect of label bias that arises from conditioning on partial input (Section 2.3) during decoding on optimization of the models. The unidirectional encoder based tagging experiments suggest that conditioning on partial input during decoding results in poor models when trained with cross entropy based methods. Interestingly, all techniques improve upon this: (i) search-aware locally and globally normalized models are able to train for accuracy directly and eliminate exposure bias that arises out of the mismatch between train-time and test-time prediction methods, and, (ii) the bidirectional tagging model which provides access to all of the input is powerful enough to learn a complex relationship between the decoder and the input representations for the search space of the CCG supertagging task and results in a much better performance.

Initialization of search-aware training

Next, we analyze the importance of appropriate initialization of search-aware optimization with pretrained models. Across all the results in Tables 3.2, 3.3 and 3.4, we observe that search-aware optimization for locally normalized models always improves upon the pre-trained locally normalized models used for initialization. But when the search-aware optimization for globally normalized models is initialized with locally normalized CE models, the improvement is not as pronounced and in the case of MT, the performance is actually *hurt* by the improper initialization for training globally normalized models – probably a consequence of large search space associated with MT and incompatibility between unnormalized scores for search-aware optimization and locally normalized scores of the *CE* model used for pre-training. When the *self-normalized* models are used for initialization, optimization for globally normalized models always improves upon the pre-trained self-normalized model. It is interesting to note that we see improvements for the globally normalized models even when $logZ$ is not exactly reduced to 0 indicating that the scores used for search-aware training initially are comparable to the scores of the pre-trained self-normalized model. We also observe that self-normalized models perform slightly worse than CE-trained models but search aware training for globally normalized models improves the performance significantly.

Search-aware training

Next, we analyze the effect of search-aware optimization on the performance of the models. Search-aware training with locally normalized models improves the performance significantly in *all* our experiments which indicates that accounting for exposure bias and optimizing for predictive performance directly is important. We also observe that the bidirectional model for tagging is quite powerful and seems to account for both *exposure bias* and *label bias* to a large extent. We reckon that this may be because the greedy decoding itself is very close to *exact search* for this well-trained tagging model over a search space that is much simpler than that associated with MT. Therefore, the impact of search-aware optimization on the bidirectional tagger is marginal. However, it is much more pronounced on the task of MT.

Global normalization and label bias

We analyze the importance of training globally normalized models. In the specific setup for tagging with the unidirectional encoder, globally normalized models are actually *more expressive* than the locally normalized models (Andor et al., 2016) as described in Section 2.3 and this is reflected in our experiments (table 3.3) with tagging. The globally normalized model (warm started with a self-normalized model) performs the best among all the models in the unidirectional tagger case which indicates that it is ameliorating something beyond exposure bias which is fixed by the search-aware locally normalized model.
For MT (table 3.4), both globally normalized and locally normalized models are equally expressive in theory because the decoder is conditioned on the full input information at each step, but we still observe that the globally normalized model improves significantly over the self-normalized pre-trained model and the search-aware locally normalized model. This indicates that it might be ameliorating the label bias associated with inexact search (discussed in Section 2.5). As discussed in Section 3.2, the globally normalized model, when initialized with a CE trained model, performs worse because of improper initialization of the search aware training. The self-normalized model starts off 1 BLEU point worse than the CE model point but global normalization, initialized with the self-normalized model improves the performance and is competitive with the best model for MT. This suggests that a better technique for initializing the optimization for globally normalized models should be helpful in improving the performance.

Global normalization and sentence length

In tables 3.5 and 3.6, we analyze the source of improvement from global normalization for MT. In table 3.5, we report the ngram overlap scores and ratio of length of the predictions to length of hypothesis for the case when the search-aware training is initialized with a self-normalized model. We observe that the globally normalized model produces longer predictions than the locally normalized model. More interestingly, it seems to have better 3 and 4-gram overlap and slightly worse unigram and bigram overlap score than the locally normalized model. These observations

	N-gram overlap	Length ratio
pretrain-beam	63.5/35.7/21.8/13.7	0.931
locally-normalized	66.9/39.4/22.7/14.0	0.918
globally-normalized	65.0/39.1/23.2/14.7	0.959

Table 3.5: Breakdown of BLEU results on de-en Machine Translation dev set. Reported on Self-normalized initialization

Src sent-length →	**0-20**	**20-30**	**30-40**	**40+**
pretrain-beam	29.36	25.73	24.71	24.50
locally-normalized	32.35	26.95	25.39	25.2
globally-normalized	33.21	28.08	26.75	26.41

Table 3.6: BLEU scores with different length inputs on dev set Reported on Self-normalized initialization. The header specifies the range of length of the input sentences

suggest that globally normalized models are better able to take longer range effects into account and are also cautious about predicting the *end-of-sentence* symbol too soon. Moreover, in table 3.6, we observe that globally normalized models perform better on all the length ranges but especially so on long sentences.

Chapter 4

Energy Based Neural Models: Scoring Non-sequential Structured Outputs

Recent work (Holtzman et al., 2019; Welleck et al., 2019b; Stahlberg and Byrne, 2019; Kumar and Sarawagi, 2019; Murray and Chiang, 2018) has suggested that locally normalized generative sequence models suffer from calibration issues and generally result in poorly trained models that assign higher scores to ill formed sentences and empty strings than natural sentences. This deficiency can directly be attributed to step-by-step locally normalized building of the sequence hypothesis that is unable to account for the future and revise scores based upon the favorable/unfavorable conditions for continuation encountered in future. While this problem is more extreme in open-ended generation, even conditional generation as is the case for machine translation also suffers from these issues as discussed in the previous section on the importance of training globally normalized sequence models.

In the previous chapters, I discussed globally normalized neural sequence networks with inference and training procedures that are essentially sequential at token level in nature. This refers to training procedures that involve decomposition of global energy computation into local sub-structure computations. Previous section has described a beam search based algorithm for training globally normalized models that aims to ameliorate some of the abovementioned problems related to label and exposure bias, however the step by step generation method with inexact search based on pruning might still suffer from issues related to calibration and assigning higher scores to ill-formed sentences. This section focuses on techniques to train and inference of globally normalized energy networks that take into account huge chunks of sequences (ideally whole sequences) at once to assign scores to the sequences via an *energy function*. Specifically, the goal of maximizing the likelihood of the observed data becomes minimizing the energy of training data containing observed outputs. The expressiveness of the distributions that can be defined over the output space depends on the parametrization of the energy function. We are interested in the case when this parametrization is defined by a neural network and henceforth, we call these neural networks *Energy Networks*. The energy network typically specifies a score for each possible output $\mathbf{y}$ for a given input $\mathbf{x}$. The parameters are shared across all the input instances and hence the training of these networks is typically amortized across the training examples.

Also in this thesis, we propose a strategy to adapt the expressiveness of the energy network

according to the complexity of the input $\mathbf{x}$ and the true distribution over $\mathcal{Y}$. This instance-wise adaptation strategy involves parametrization of the evolution of the hidden layers of the network along continuous "depth" of the network via *neural ordinary differential equations*. This provides an opportunity to tailor the depth of the neural energy network according to the complexity of modeling the input and should arguably be better than a fully amortized parametrization of the energy network.

However, as we discus later energy functions parametrized by neural networks are typically continuous and smooth. This results in easier optimization for modelling data that has a continuous domain like images. Discrete domains involved with text and sets lend themselves to energy networks with more difficulty and we have fewer options to optimize the energy functions to fit discrete domains. Hence, in this thesis, before discussing modelling discrete token sequences via flexible energy networks, we will first propose algorithms and techniques to train energy networks on images. This would also shed light on the performance of *adaptive* energy functions in modelling objects from a continuous domain which is more suitable for these neural energy networks. To draw a contrast between the challenges associated with learning flexible energy functions over continuous domains and discrete domains, this thesis will consider the problem of learning global energy functions over discrete sets. More concretely, we will use energy function to model interdependence between potential individual members of a set, where the *power set* characterizes all the possible outcomes that need to be summed over to estimate the partition function. This problem is particularly interesting because the individual items do not have an intrinsic order in which they appear in a set which makes this task a very different one from sequence modelling where there are strong ordering constraints between the tokens. Multi label classification tasks are an example for conditional set modelling. However, several interesting unconditional set modelling problems include tasks like ingredient modeling in cooking recipes where the individual ingredients have possible n-ary potential relationships.

Finally, we explore globally normalized scoring models for discrete sequences that instead of building the scores step-wise in a left-to-right direction (generally with pruning at every step), focus on assigning scores to whole sequences at once. We will specifically focus on robustness of these models and their ability to assign low scores to bad sequences that involve miscalibrated *end-of-sequence* token probabilities, repetition of tokens and ngrams etc. The approaches this thesis proposes to consider explicitly relates to the models abandoning the left-to-right generation/scoring paradigm and focuses heavily on unordered generation of order sensitive sequences to assign scores to sequences. Before we discus the specific tasks, we will briefly review *Structured Prediction Energy Networks* (Belanger and McCallum, 2016) and Neural Ordinary Differential Equations (Chen et al., 2018) which form the inspiration behind the methods proposed in this thesis.

4.1 Structured Prediction Energy Networks

As mentioned in the introduction, *Structured Prediction Energy Networks* (Belanger and McCallum, 2016) associate a score or energy $E(\mathbf{y})$, with every possible $\mathbf{y} \in \mathcal{Y}$ computed by a scoring function $f(\mathbf{y}; \theta)$ that is parametrized via a neural network with parameters θ. Moreover for each input $\mathbf{x}$, a different distribution over $\mathcal{Y}_{\mathbf{x}}$ needs to be induced so the energy networks also take in $\mathbf{x}$ as input and

the energy function is hence denoted as $E(\mathbf{y}, \mathbf{x}; \theta)$. The probability distribution over $\mathcal{Y}$ is defined via the energy function as:

$$p(\mathbf{y}) = \frac{\exp\left(-E(\mathbf{y}; \theta)\right)}{Z}$$

where $Z = \sum_{\mathbf{y} \in \mathcal{Y}} \exp\left(-E(\mathbf{y}; \theta)\right)$ and for continuous spaces, $Z = \int_{y \in \mathcal{Y}} \exp\left(-E(\mathbf{y}; \theta)\right) dy$. More specifically, for each input $\mathbf{x}$, the distribution is defined as:

$$p(\mathbf{y} \mid \mathbf{x}) = \frac{\exp\left(-E(\mathbf{y}, \mathbf{x}; \theta)\right)}{Z(\mathbf{x})}$$

where $Z(\mathbf{x}) = \sum_{\mathbf{y} \in \mathcal{Y}} \exp\left(-E(\mathbf{y}, \mathbf{x}; \theta)\right)$. The energy function can be flexibly defined over the global structured representation of $\mathbf{y}$, specific cliques of $\mathbf{y}$ and a representation for the input $\mathbf{x}$ via common neural network architectures like multi-layer perceptrons (MLPs), autoregressive neural representations or convolutional networks. For finding the highest scoring $\mathbf{y}$ for discrete problems the objective could be written as:

$$arg\ min_{\mathbf{y} \in \mathcal{Y}} E(\mathbf{y}, \mathbf{x}; \theta) \tag{4.1}$$

However optimizing this problem is difficult and since the energy based function is parametrized by a neural network, for ease of optimization, the discrete space $\mathcal{Y}$ is relaxed to a continuous space $\bar{\mathcal{Y}}$.

$$arg\ min_{\bar{\mathbf{y}} \in \bar{\mathcal{Y}}} E(\bar{\mathbf{y}}, \mathbf{x}; \theta) \tag{4.2}$$

where $\bar{\mathbf{y}}$ is the *relaxed* structure output that might not correspond to any of the entries in the discrete output space. Therefore, when working in this setting, an important component of the pipeline is to project the relaxed $\bar{\mathbf{y}}$ to the valid discrete output space $\mathcal{Y}$. This issue with training is non-existent in the case of continuous output spaces and hence relaxation is not required and $\bar{\mathbf{y}} = \mathbf{y}$. The optimization over the continuous output space can be performed via gradient descent, where the gradients are taken with respect to $\bar{\mathbf{y}}$ instead of the model parameters

An energy network can be trained in a straightforward manner using maximum likelihood if enumeration/integration over $\mathcal{Y}$ is tractable. However, this is not the case in most problems of interest and hence training approaches focus on either approximation of the likelihood or surrogate discriminative objectives.

4.2 Neural Differential Equations

As mentioned in the previous section, we propose a technique for flexible energy networks that adapt their expressiveness according to the complexity of the observed data instances. This technique is inspired by neural differential equations (Chen et al., 2018) which view the different hidden layers in the architecture as evaluations of a single hidden state that evolves continuously with depth (potentially infinitely) whose evolution is governed by a differential equation. Neural differential equations involve a neural parametrization of this evolution function and the goal of learning is to learn the parameters for the differential equation. The parametrized differential equations endow

the network with arbitrary modelling capacity with infinite depth, but with a finite set of parameters. But more importantly, these networks provide a way to adapt the depth of the energy network with complexity of the input instance. More concretely, in a neural network let the hidden state at a particular depth t be denoted by $\mathbf{h}(t)$, then the evolution of the state is governed by a parametrized equation (with parameters θ):

$$\frac{d\mathbf{h}(t)}{dt} = f(\mathbf{h}(t), t, \theta)$$

where f is the evolution function that we desire to learn in order to best fit the data and f can be expressed as a neural network. In order to train these models endowed with evolution of the hidden state specified by a differential equation, access to numerical solvers and their gradients is required for backpropagation based learning. Chen et al. (2018) propose an approach that treats the forward pass of the solver as a black box and computes the gradients via an adjoint sensitivity method which results in a solving a related but different differential equation backwards in time. An interesting perspective related to this parametrization is that if *Euler's method* is used as a solver, then the updates to the hidden states mimic the RESNET (He et al., 2016) structure with tied parameters across layers. The *adjoint* ($\mathbf{a}$) required for computing the gradient of final loss(L) wrt. parameters is basically the gradient of L wrt. the hidden state i.e. $\mathbf{a}(t) = \frac{\partial L}{\partial \mathbf{h}(t)}$, and the evolution of the adjoint with depth t is given by the equation:

$$\frac{\partial \mathbf{a}(t)}{\partial t} = -\mathbf{a}(t)^T \frac{\partial f(\mathbf{h}(t), t, \theta)}{\partial \mathbf{h}}$$

This differential equation basically is a substitute for instantaneous chain rule in back propagation for the continuously evolving hidden state. The form of full gradient wrt. parameters can be derived via chain rule using the above gradient of the adjoint that involves solving a differential equation. Hence, both the forward and backward passes of Neural ODEs involve using a numerical solver for the ODEs. As discussed in the next section, using an adaptive step size solver like *Adams-Bashforth method* results in a procedure which adapts the effective depth of the network and computational complexity according to the complexity of modeling the given input instance.

4.2.1 Instance-wise adaptation of the complexity of the Energy Network

As noted above, the choice of numerical solver for solving the differential equations affects the flexibility of computation and accuracy of solution. Using Euler's method for the parametrization of evolution of the hidden state results in an architecture that is basically a resnet with weight tying at each layer. The number of layers in this setting is the hyperparameter specifying the number of evaluation steps in the Euler's method. This method is inexact and has poor at estimating the numerical antegration solution. An adaptive step-size method like Adams-Bashforth method is more powerful and accurate. Moreover, it also belongs to a family of solvers that allow a parametrization thacan adapt itself to the complexity of the input and the initial hidden state. For example the learned parametrization might more easily approximate the integrand with fewer evaluation steps at certain points in the domain of the hidden states and hence result in a smaller implicit 'depth' of the

network but might require a lot more small steps for accurately modeling the evolution of a more complex initial hidden state and hence have a larger effective depth. This work uses this intuition to propose flexible energy networks parametrized by a neural ODE which automatically adapt the effective depth according to the complexity of $\mathbf{x}$ and $\mathbf{y}$.

4.3 Energy Networks for continuous domains: Image generation

Generative modeling of images has received a lot of attention with the prevalent models exhibiting good performance at either of the two important paradigms pertinent to image generation:

- **Density Estimation**: Locally normalized autoregressive models like PixelCNN (Van den Oord et al., 2016) achieve good likelihood performance over popular image datasets but are expensive to train and sample from due to poor parallelizability. Moreover, they often result in poor samples. VAE (Kingma and Welling, 2013) based models (Gulrajani et al., 2016; Gregor et al., 2015) are faster to train and also achieve good likelihoods, however they tend to suffer from the optimization problem related to latent variable collapse and end up relying on the decoder (which can be as powerful as an autoregressive network or as weak as independent Gaussian emissions at each pixel) for likelihood maximization. This class of models is also known to generate blurry samples.
- **Realistic Sample Generation**: Flow based models like RealNVP (Dinh et al., 2016) and Glow (Kingma and Dhariwal, 2018) allow exact inference and generate realistic samples, however they are poor at maximizing the likelihoods primarily due to the limitations imposed on their parametrization for efficient training/inference. They require the input transformations to be invertible and this imposes constraints on the kinds of transformations that a model allows for. GANs (Goodfellow et al., 2014) are another method to train generative models of images (Zhao et al., 2016) with good samples but the optimization tends to be unstable during training. Also being an implicit generative model, they don't readily lend themselves to likelihood based computations.

The above described classes of models are not good at both of these paradigms. This thesis explores the possibility of a newer class of energy based generative models of images that have explicit likelihood associated with them while also being able to generate good samples. This has been nascently explored recently (Du and Mordatch, 2019) with convolutional parametrization of the energy function that is trained by 1-sample approximation of the log-normalizer with the sample being generated via Hamiltonian Monte Carlo method.

More concretely, we are interested in maximizing the likelihood of a set of images in the training data $\mathbf{x}_{tr}$ i.e. samples drawn from the true distribution p_d, using a generative model parametrized by an energy network that assigns a score/energy $E(\mathbf{x};\theta)$ value to an image $\mathbf{x}$. The Boltzmann distribution associated with this network for an image $\mathbf{x}$ is $p(\mathbf{x}) = \frac{\exp\left(-E(\mathbf{x};\theta)\right)}{Z(\theta)}$ where $Z(\theta) = \int_{x\in\mathcal{X}} \exp\left(-E(\mathbf{x};\theta)\right)dx$ and the objective is to maximize the log-likelihood:

$$\min \mathcal{L}_{ML} = \mathbb{E}_{\mathbf{x}\sim p_d} E(\mathbf{x};\theta) - log\ Z(\theta)$$

Since the gradient computation involves an expectation over the model distribution due to the log-normalizer Z which is expensive to compute, it is common to approximate the quantities associated with the Z term with Monte Carlo samples. For example, the gradient can be written as:

$$\nabla_\theta \mathcal{L}_{ML} = \mathbb{E}_{\mathbf{x}^+ \sim p_d}[\nabla_\theta E(\mathbf{x}^+; \theta)] - \mathbb{E}_{\mathbf{x}^- \sim p_\theta}[\nabla_\theta E(\mathbf{x}^-; \theta)]$$

The first term is easy to compute but the second term involving an expectation over the model distribution is typically approximated via a Monte Carlo estimate based on samples. Previous work (**?**) has explored a 1-sample approximation where the sample is drawn via k-steps of langevin dynamics which is a Monte Carlo sampling method based upon using the gradient information of the energy function wrt. $\mathbf{x}$ in the transition function to suggest the next location to evaluate in the Monte Carlo chain. This method yielded decent results but was found to be difficult to optimize due to high variance of the Langevin dynamics. Although, Langevin dynamics often results in faster mixing compared to other monte carlo sampling methods, it tends to be difficult to optimize with because of high sensitivity to the hyperparameters like leap-frog step size etc. associated with the sampling method. In this thesis, we will explore different methods of training the energy network for images which include:

1. **Contrastive divergence and Persistent Contrastive divergence**: In previous work on learning Restricted Boltzmann Machines (Salakhutdinov and Hinton, 2009), contrastive divergence (CD) (Hinton, 2002) has shown to be an effective method for generating the negative samples. It involves starting from a real sample in the training data and running the Markov chain for a small number of steps (typically 1), to generate a negative example. While the theoretical properties of this method are underexplored, empirically it has been shown to be biased (Carreira-Perpinan and Hinton, 2005) when compared to true maximum likelihood estimation. Perisistent contrastive divergence (Tieleman and Hinton, 2009) involves keeping a buffer of generated samples throughout the training procedure using contrastive divergence so that the effect of running the Monte Carlo chain for k steps is reflected in the training procedure, however intermittent updates to the energy function result in it being an improper sampling method which is not guaranteed to match the original distribution. Moreover, training with PCD often results in heavily autocorrelated negative samples.
2. **Optimizing Energy Network via gradient steps**: This focuses on approximating the partition function via a few low energy sample representatives. This approach involves finding a local energy minimum for estimation of the normalizer and one method to do this for smooth energy functions over a continuous domain would be to perform gradient ascent with respect to the input variable $\mathbf{x}$ in order to find low energy samples. This approach resemebles the training procedure proposed in SPENs (Belanger et al., 2017).
3. **Learning an auxilliary/proposal distribution**: This approach focuses on training a tractable proposal distribution which is close to the distribution represented by the energy network and can be used to approximate the log normalizer and other quantities related to the log normalizer via techniques like importance sampling, likelihood ratio estimators, lower bound maximization etc. This method is appealing because it lets us capture the desirable properties of other kinds of generative models like autoregressive generative models into the optimization of the energy network via the proposal distribution.

As mentioned in the previous section, most importantly, this work will focus on Energy networks parameterized by *Neural ODEs* trained with adaptive step size methods so that the energy network can flexibly adapt to the complexity of modeling different kinds of images.

4.4 Energy Networks for discrete domains

Although, the core principles related to assigning a score based on a continuously parametrized energy function to each possible hypothesis over the domain are shared with energy networks over continuous domains, generative modeling of discrete entities inherently is a very different problem. Several training techniques that have been successful in training energy networks for modeling artefacts with continuous domains like images are expected to fail on training energy network based generative models for objects with discrete domains because of unavailability of associated gradients. Hence, gradient based SPEN optimization or samplign based on Langevin dynamics is unlikely to work for discrete problems directly without a mechanism for projection of discrete entities onto a continuous space. To compound the difficulty further, this projection is non-trivial to find because typically, not even a natural partial ordering exists over the individual discrete entities. However, a key difference that makes reasoning over discrete entities different is that for most problems of interest, the hypothesis space is *countable* which suggests that discrete sampling methods are naturally suitable for approximation for the Energy networks' distribution. This also opens up the possibilities to consider techniques from combinatorics. This thesis proposes to develop training and inference techniques for globally normalized energy networks to model sets and sequences.

As discussed in the previous section, we are interested in the parametrization of the energy network with a neural ODE such that the network is able to account for the complexity of the input.

4.4.1 Discrete set prediction

As discussed earlier, modeling sets in a globally normalized manner is interesting because the problem requires discovering n-ary affinities between potential discrete members of a set and the order in which these members appear doesn't matter. Multi label classification is a well explored conditional set prediction problem (Yu et al., 2014; Hsu et al., 2009). Most of these approaches either assume an ordering for set prediction or train over a few different orderings of the members in the training data sets. Recently, work on deep-sets (Zaheer et al., 2017) considers the permutation invariance between the members of the sets and proposes a neural activation function and architecture that is invariant to the ordering of the input sets to make predictions on the basis of the input sets. The key difference between deep-sets and the proposed work in this thesis is that deep-sets are suitable for supervised learning problems that take sets as inputs whereas we are interested in probabilistic generative modeling of sets themselves in a permutation invariant manner. The problem setup consists of an inventory of discrete items that could potentially make up the sets denoted by $\mathcal{V}$. In this setting we describe a set via a binary vector of length matching the inventory size, $\{0, 1\}^{|\mathcal{V}|}$, where 1 indicates the presence of an item in the set. The task is to model the interdependence between the items in $\mathcal{V}$ such that they explain the observed sets in the world.

This problem can be conditional as it is in the case of multi label classification, or this could be unconditional as it is in the case of modeling ingredients in a recipe, hardware parts in automobiles etc.

To test the efficacy of energy networks in modelling the potential between the potential members of a set, we propose to model unconditional set generation. An example of this setting would be to model ingredients in various recipes. Given an inventory of ingredients, the task is to pick the ingredients that combine well. This is a task capable of representing rich interactions between the members of the inventory. We plan to use the dataset used by (Kiddon et al., 2016) which includes about 83000 recipes.

We plan to use global factors via an energy network and also the permutation invariant functions explored in work related to deep-sets to assign energy to each possible set. Exhaustive enumeration of the power set is typically not feasible, therefore we'll use approximations to the log-partition function via sample-based methods. We'll also leverage order sensitive methods to aid in the optimization of the global model by potentially training it as a reranker for the candidates generated by an order sensitive LSTM based model of sets.

4.4.2 Discrete sequence modeling

As discussed earlier, step-by-step left-to-right building of the sequence hypothesis with pruning that is unable to account for the future and revise scores arbitrarily based upon the favorable/unfavorable states for continuation results in generative models that are poorly calibrated and assign high scores to ill formed sequences. While this problem is more extreme in open-ended generation, even conditional generation as is the case for machine translation also suffers from these issues. This chapter focuses on proposing models that aim to assign scores to whole sequences at once and ideally do not make pruning and scoring decisions based upon partial completions of sequences.

The approaches this thesis proposes to consider explicitly relates to the models abandoning the left-to-right generation/scoring paradigm and focuses heavily on unordered generation of order sensitive sequences to assign scores to sequences. Another important motivation for abandoning the left-to-right sequential generation is that it would help in making the density estimation operations parallelizable and hence faster to train on text datasets. These approaches include:

1. **Incorporating left and right contexts**: Recently, there have been attempts (Wang and Cho, 2019) to interpret BERT (Devlin et al., 2018) as a probabilistic generative model of text in which the training is done via pseudolikelihood maximization. We propose to use transformer architectures used in BERT and XLNet (Yang et al., 2019) to assign scores to sequences which are informed by the whole context of each word.
2. **Unordered sequence scoring**: This paradigm focuses on generating the length of the sequence first and generating the words in an arbitrary order (Welleck et al., 2019a) such that subsequent word predictions condition themselves on previously predicted words and their positions. This paradigm is inspired by previous work on easy first tagging/parsing (Tsuruoka and Tsujii, 2005; Martins and Kreutzer, 2017; Ma et al., 2014).

Sequential models as auxilliary/Proposal distribution: While the aim of training globally normalized models is desirable, it is difficult in practice due to the challenges associated with

non-convex optimization and intractability of the log-partition function. Hence, we propose to use existing sequential generative models of text, which despite their problems have been impressive as generative models for text in many aspects, to aid the optimization of the globally normalized models. As discussed earlier in the context of training generative models for images, we propose to use the commonly prevalent left-to-right sequential generative models as proposal distribution which can lend themselves well to importance-sampling based training methods or approximating the globally normalized likelihood via a lower bound with the sequence distribution as the proposal distribution.

Another straightforward method to leverage sequential left-to-right models to train globally normalized models would be to generate high scoring candidates via the auxilliary sequential distribution and training the globally normalized energy network as a *reranker* of the candidates. This approach might still involve pruning of better candidates but training the globally normalized energy function as a reranker is expected to be a better initialization point for more complex methods for training the global energy networks.

Chapter 5

Proposed Timeline

- November 2019: Work on print and probability and get a paper on super-resolution ready for ACL-2020 (work not mentioned in the thesis)
- December–February 2019: Work on energy networks for sets (aim for ICML 2020/EMNLP 2020)
- February–May 2020: Work on energy networks for images (aim for NeurIPS 2020/ICML 2020)
- May–July 2020: Energy networks for sequences. submit work.
- August 2020: Anomaly detection/OCR work on printed text and handwriting for print and probability project (work not mentioned in the thesis)
- August–October 2020: Conduct job search.
- September–November 2020: Run BSO (Wiseman and Rush, 2016) comparisons described in 2.4.5 and write thesis.
- November–December 2020: Defend thesis.

Chapter 6

Conclusion

This document describes my work on training locally normalized and globally normalized neural sequence models. More specifically, this document describes our work (Goyal et al., 2017a, 2018) on search-aware training methods that are amenable to backpropagation for locally normalized neural sequence models that ameliorate the issues like exposure bias pertinent to common training methods like teacher forcing. The approaches to achieve this at their core, involved continuous relaxation to discontinuous inference procedures like greedy decoding, sampling, and beam search so that these procedures could be incorporated into backpropagation based training procedures which rely on existence of gradients and subgradients. Furthermore, this work (Goyal et al., 2019) also describes our work on training globally normalized models using a continuous relaxation to beam search for sequences which empirically outperform their comparable locally normalized counterparts. We attribute this to label bias experienced by locally normalized models during search-aware training. Having established the need to explore globally normalized models for globally normalized models for discrete sequences, this document proposes further work on energy-based globally normalized models.

Specifically, during rest of my Ph.D., I propose to develop better globally normalized models via an energy network based parametrization, that are flexible, robust and computationally efficient for a variety of domains including discrete sequences, images and discrete sets. For sequences, we aim to abandon the paradigm of left-to-right stepwise construction of scores and instead focus on more holistic and order agnostic methods for computation of energy for the sequences. In order to understand the sensitivity of energy models to representative domains, we also propose to develop models for continuous domain objects like images which allow for gradient based inference, and discrete sets, which like sequences deal with discrete objects but are invariant to permutations of the constituent elements.

Bibliography

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *Association for Computational Linguistics*. 6, 26, 28, 34

Jacob Andreas and Dan Klein. 2015. When and why are log-linear models self-normalizing? In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. pages 244–249. 29

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. An actor-critic algorithm for sequence prediction. In *International Conference on Learning Representations*. 6

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*. 11

Dzmitry Bahdanau, Dmiriy Serdyuk, Philémon Brakel, Nan Rosemary Ke, Jan Chorowski, Aaron Courville, and Yoshua Bengio. 2016. Task loss estimation for structured prediction . 32

David Belanger and Andrew McCallum. 2016. Structured prediction energy networks. In *International Conference on Machine Learning*. pages 983–992. 37

David Belanger, Bishan Yang, and Andrew McCallum. 2017. End-to-end learning for structured prediction energy networks. *arXiv preprint arXiv:1703.05667* . 41

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*. pages 1171–1179. 2, 6, 7, 8, 10, 11, 12, 13, 21

Miguel A Carreira-Perpinan and Geoffrey E Hinton. 2005. On contrastive divergence learning. In *Aistats*. Citeseer, volume 10, pages 33–40. 41

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th iwslt evaluation campaign, iwslt 2014. In *Proceedings of the International Workshop on Spoken Language Translation, Hanoi, Vietnam*. 11, 32

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. 2018. Neural ordinary differential equations. In *Advances in neural information processing systems*. pages 6571–6583. 3, 37, 38, 39

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On

the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* . 6, 13

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, page 111. 28

Hal Daumé, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. *Machine learning* 75(3):297–325. 6, 10

Hal Daumé III and Daniel Marcu. 2005. Learning as search optimization: Approximate large margin methods for structured prediction. In *Proceedings of the 22nd international conference on Machine learning*. ACM, pages 169–176. 6

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* . 43

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. volume 1, pages 1370–1380. 29

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. 2016. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803* . 40

Yilun Du and Igor Mordatch. 2019. Implicit generation and generalization in energy-based models. *arXiv preprint arXiv:1903.08689* . 40

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. pages 2672–2680. 40

Matthew R. Gormley, Mark Dredze, and Jason Eisner. 2015. Approximation-aware dependency parsing by belief propagation. *Transactions of the Association for Computational Linguistics (TACL)* . 6

Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. 2017a. Differentiable scheduled sampling for credit assignment. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. volume 2, pages 366–371. 2, 46

Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. 2017b. Differentiable scheduled sampling for credit assignment. *arXiv preprint arXiv:1704.06970* . 17

Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. 2019. An empirical investigation of global and local normalization for recurrent neural sequence models using a continuous relaxation to beam search. In *Proceedings of 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 2, 46

Kartik Goyal, Graham Neubig, Chris Dyer, and Taylor Berg-Kirkpatrick. 2017c. A continuous relaxation of beam search for end-to-end training of neural sequence models. *arXiv preprint arXiv:1708.00111* . 27, 28, 31

Kartik Goyal, Graham Neubig, Chris Dyer, and Taylor Berg-Kirkpatrick. 2018. A continuous relaxation of beam search for end-to-end training of neural sequence models. In *Proceedings of AAAI Conference on Artificial Intelligence*. 2, 46

Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. 2015. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623* . 40

Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taiga, Francesco Visin, David Vazquez, and Aaron Courville. 2016. Pixelvae: A latent variable model for natural images. *arXiv preprint arXiv:1611.05013* . 40

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. pages 297–304. 29

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. pages 770–778. 39

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* . 30

Geoffrey E Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural computation* 14(8):1771–1800. 41

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780. 16

Julia Hockenmaier and Mark Steedman. 2002. Acquiring compact lexicalized grammars from a cleaner treebank. In *LREC*. 21, 31

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751* . 2, 36

Daniel J Hsu, Sham M Kakade, John Langford, and Tong Zhang. 2009. Multi-label prediction via compressed sensing. In *Advances in neural information processing systems*. pages 772–780. 42

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*. 9, 16

Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. 2016. Globally coherent text generation with neural checklist models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. pages 329–339. 43

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* . 40

Durk P Kingma and Prafulla Dhariwal. 2018. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*. pages 10215–10224. 40

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872* . 6, 13

Aviral Kumar and Sunita Sarawagi. 2019. Calibration of encoder decoder models for neural machine

translation. *arXiv preprint arXiv:1903.00802* . 36

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data . 26

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. 2006. A tutorial on energy-based learning. *Predicting structured data* 1(0). 2

Ji Ma, Yue Zhang, and Jingbo Zhu. 2014. Tagging the web: Building a robust web tagger with neural network. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. pages 144–154. 43

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*. 9, 16

André FT Martins and Julia Kreutzer. 2017. Learning what's easy: Fully differentiable neural easy-first taggers. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. pages 349–362. 43

Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. *arXiv preprint arXiv:1206.6426* . 29

Kenton Murray and David Chiang. 2018. Correcting length bias in neural machine translation. *arXiv preprint arXiv:1808.10006* . 36

Graham Neubig. 2017. Neural machine translation and sequence-to-sequence models: A tutorial. *arXiv preprint arXiv:1703.01619* . 6, 13

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *International Conference on Learning Representations*. 6, 11, 21, 32

Stéphane Ross, Geoffrey J Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*. volume 1, page 6. 6, 10

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Empirical Methods in Natural Language Processing*. 21

Ruslan Salakhutdinov and Geoffrey Hinton. 2009. Deep boltzmann machines. In *Artificial intelligence and statistics*. pages 448–455. 41

Noah A Smith and Mark Johnson. 2007. Weighted and probabilistic context-free grammars are equally expressive. *Computational Linguistics* 33(4):477–491. 27

Felix Stahlberg and Bill Byrne. 2019. On nmt search errors and model errors: Cat got your tongue? *arXiv preprint arXiv:1908.10090* . 36

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. pages 3104–3112. 2, 15

Ben Taskar, Carlos Guestrin, and Daphne Koller. 2004. Max-margin markov networks. In *Advances in neural information processing systems*. pages 25–32. 18

Tijmen Tieleman and Geoffrey Hinton. 2009. Using fast weights to improve persistent contrastive

divergence. In *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, pages 1033–1040. 41

Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics, pages 142–147. 11, 21

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large margin methods for structured and interdependent output variables. *Journal of machine learning research* 6(Sep):1453–1484. 18

Yoshimasa Tsuruoka and Jun'ichi Tsujii. 2005. Bidirectional inference with the easiest-first strategy for tagging sequence data. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*. Association for Computational Linguistics, pages 467–474. 43

Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. 2016. Conditional image generation with pixelcnn decoders. In *Advances in neural information processing systems*. pages 4790–4798. 40

Ashish Vaswani, Yonatan Bisk, Kenji Sagae, and Ryan Musa. 2016. Supertagging with lstms. In *Proceedings of NAACL-HLT*. pages 232–237. 21, 31

Alex Wang and Kyunghyun Cho. 2019. Bert has a mouth, and it must speak: Bert as a markov random field language model. *arXiv preprint arXiv:1902.04094* . 43

Sean Welleck, Kianté Brantley, Hal Daumé III, and Kyunghyun Cho. 2019a. Non-monotonic sequential text generation. *arXiv preprint arXiv:1902.02192* . 43

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019b. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319* . 2, 36

Sam Wiseman and Alexander M Rush. 2016. Sequence-to-sequence learning as beam-search optimization. In *Empirical Methods in Natural Language Processing*. 6, 20, 24, 26, 28, 45

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237* . 43

Hsiang-Fu Yu, Prateek Jain, Purushottam Kar, and Inderjit Dhillon. 2014. Large-scale multi-label learning with missing labels. In *International conference on machine learning*. pages 593–601. 42

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan R Salakhutdinov, and Alexander J Smola. 2017. Deep sets. In *Advances in neural information processing systems*. pages 3391–3401. 42

Junbo Zhao, Michael Mathieu, and Yann LeCun. 2016. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126* . 40