# Bandit Structured Prediction for Neural Sequence-to-Sequence Learning

**Julia Kreutzer*** and **Artem Sokolov*** and **Stefan Riezler†,***
*Computational Linguistics & †IWR, Heidelberg University, Germany
{kreutzer,sokolov,riezler}@cl.uni-heidelberg.de

## Abstract

Bandit structured prediction describes a stochastic optimization framework where learning is performed from partial feedback. This feedback is received in the form of a task loss evaluation to a predicted output structure, without having access to gold standard structures. We advance this framework by lifting linear bandit learning to neural sequence-to-sequence learning problems using attention-based recurrent neural networks. Furthermore, we show how to incorporate control variates into our learning algorithms for variance reduction and improved generalization. We present an evaluation on a neural machine translation task that shows improvements of up to 5.89 BLEU points for domain adaptation from simulated bandit feedback.

## 1 Introduction

Many NLP tasks involve learning to predict a structured output such as a sequence, a tree or a graph. Sequence-to-sequence learning with neural networks has recently become a popular approach that allows tackling structured prediction as a mapping problem between variable-length sequences, e.g., from foreign language sentences into target-language sentences (Sutskever et al., 2014), or from natural language input sentences into linearized versions of syntactic (Vinyals et al., 2015) or semantic parses (Jia and Liang, 2016). A known bottleneck in structured prediction is the requirement of large amounts of gold-standard structures for supervised learning of model parameters, especially for data-hungry neural network models. Sokolov et al. (2016a,b) presented a framework for stochastic structured prediction under bandit feedback that alleviates the need for labeled output structures in learning: Following an online learning protocol, on each iteration the learner receives an input, predicts an output structure, and receives partial feedback in form of a task loss evaluation of the predicted structure.[1] They "banditize" several objective functions for linear structured predictions, and evaluate the resulting algorithms with simulated bandit feedback on various NLP tasks.

We show how to lift linear structured prediction under bandit feedback to non-linear models for sequence-to-sequence learning with attention-based recurrent neural networks (Bahdanau et al., 2015). Our framework is applicable to sequence-to-sequence learning from various types of weak feedback. For example, extracting learning signals from the execution of structured outputs against databases has been established in the communities of semantic parsing and grounded language learning since more than a decade (Zettlemoyer and Collins, 2005; Clarke et al., 2010; Liang et al., 2011). Our work can build the basis for neural semantic parsing from weak feedback.

In this paper, we focus on the application of machine translation via neural sequence-to-sequence learning. The standard procedure of training neural machine translation (NMT) models is to compare their output to human-generated translations and to infer model updates from this comparison. However, the creation of reference translations or post-edits requires professional expertise of users. Our framework allows NMT models to learn from feedback that is weaker than human references or post-edits. One could imagine a scenario of personalized machine translation where translations have to be adapted to the user's specific purpose and domain. The feedback required by our methods can be provided by laymen users or can even

[1] The name "bandit feedback" is inherited from the problem of maximizing the reward for a sequence of pulls of arms of so-called "one-armed bandit" slot machines.

be implicit, e.g., inferred from user interactions with the translated content on a web page.

Starting from the work of Sokolov et al. (2016a,b), we lift their objectives to neural sequence-to-sequence learning. We evaluate the resulting algorithms on the task of French-to-English translation domain adaptation where a seed model trained on Europarl data is adapted to the NewsCommentary and the TED talks domain with simulated weak feedback. By learning from this feedback, we find 4.08 BLEU points improvements on NewsCommentary, and 5.89 BLEU points improvement on TED. Furthermore, we show how control variates can be integrated in our algorithms, yielding faster learning and improved generalization in our experiments.

## 2 Related Work

NMT models are most commonly trained under a word-level maximum likelihood objective. Even though this objective has successfully been applied to many sequence-to-sequence learning tasks, the resulting models suffer from exposure bias, since they learn to generate output words based on the history of given reference words, not on their own predictions. Ranzato et al. (2016) apply techniques from reinforcement learning (Sutton and Barto, 1998; Sutton et al., 2000) and imitation learning (Schaal, 1999; Ross et al., 2011; Daumé et al., 2009) to learn from feedback to the model's own predictions. Furthermore, they address the mismatch between word-level loss and sequence-level evaluation metric by using a mixture of the REINFORCE (Williams, 1992) algorithm and the standard maximum likelihood training to directly optimize a sequence-level loss. Similarly, Shen et al. (2016) lift minimum risk training (Och, 2003; Smith and Eisner, 2006; Gimpel and Smith, 2010; Yuille and He, 2012; He and Deng, 2012) from linear models for machine translation to NMT. These works are closely related to ours in that they use the technique of score function gradient estimators (Fu, 2006; Schulman et al., 2015) for stochastic learning. However, the learning environment of Shen et al. (2016) is different from ours in that they approximate the true gradient of the risk objective in a full information setting by sampling a subset of translations and computing the expectation over their rewards. In our bandit setting, feedback to only a single sample per sentence is available, making the learning problem much harder. The approach by Ranzato et al. (2016) approximates the expectation with single samples, but still requires reference translations which are unavailable in the bandit setting.

To our knowledge, the only work on training NMT from weak feedback is the work by He et al. (2016). They propose a dual-learning mechanism where two translation models are jointly trained on monolingual data. The feedback in this case is a reward signal from language models and a reconstruction error. This is attractive because the feedback can automatically be generated from monolingual data and does not require any human references. However, we are interested in using estimates of human feedback on translation quality to directly adapt the model to the users' needs.

Our approach follows most closely the work of Sokolov et al. (2016a,b). They introduce bandit learning objectives for structured prediction and apply them to various NLP tasks, including machine translation with linear models. Their approach can be seen as an instantiation of reinforcement learning to one-state Markov decision processes under linear policy models. In this paper, we transfer their algorithms to non-linear sequence-to-sequence learning. Sokolov et al. (2016a) showed applications of linear bandit learning to tasks such as multiclass-classification, OCR, and chunking, where learning can be done from scratch. We focus on lifting their linear machine translation experiments to the more complex NMT that requires a warm start for training. This is done by training a seed model on one domain and adapting it to a new domain based on bandit feedback only. For this task we build on the work of Freitag and Al-Onaizan (2016), who investigate strategies to find the best of both worlds: models that adapt well to the new domain without deteriorating on the old domain. In contrast to previous approaches to domain adaptation for NMT, we do not require in-domain parallel data, but consult direct feedback to the translations generated for the new domain.

## 3 Neural Machine Translation

Neural models for machine translation are based on a sequence-to-sequence learning architecture consisting of an encoder and a decoder (Cho et al., 2014; Sutskever et al., 2014; Bahdanau et al., 2015). An encoder Recurrent Neural Network (RNN) reads in the source sentence and a decoder

RNN generates the target sentence conditioned on the encoded source.

The input to the encoder is a sequence of vectors $\mathbf{x} = (x_1, \ldots, x_{T_x})$ representing a sequence of source words of length $T_x$. In the approach of Sutskever et al. (2014), they are encoded into a single vector $c = q(\{h_1, \ldots, h_{T_x}\})$, where $h_t = f(x_t, h_{t-1})$ is the hidden state of the RNN at time $t$. Several choices are possible for the non-linear functions $f$ and $q$: Here we are using a Gated Recurrent Unit (GRU) (Chung et al., 2014) for $f$, and for $q$ an attention mechanism that defines the context vector as a weighted sum over encoder hidden states (Bahdanau et al., 2015; Luong et al., 2015a).

The decoder RNN predicts the next target word $y_t$ at time $t$ given the context vector $c$ and the previous target words $\mathbf{y}_{<t} = \{y_1, \ldots, y_{t-1}\}$ from a probability distribution over the target vocabulary $V$. This distribution is the result of a softmax transformation of the decoder outputs $\mathbf{o} = \{o_1, \ldots, o_{T_y}\}$, such that

$$p_\theta(y_t = w_i | \mathbf{y}_{<t}, c) = \frac{\exp(o_{w_i})}{\sum_{v=1}^{V} \exp(o_{w_v})}.$$

The probability of a full sequence of outputs $\mathbf{y} = (y_1, \ldots, y_{T_y})$ of length $T_y$ is defined as the product of the conditional word probabilities:

$$p_\theta(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{T_y} p_\theta(y_t | \mathbf{y}_{<t}, c).$$

Since this encoder-decoder architecture is fully differentiable, it can be trained with gradient descent methods. Given a parallel training set of $S$ source sentences and their reference translations $D = \{(\mathbf{x}^{(s)}, \mathbf{y}^{(s)})\}_{s=1}^{S}$, we can define a word-level Maximum Likelihood Estimation (MLE) objective, which aims to find the parameters

$$\hat{\theta}^{\text{MLE}} = \arg\max_\theta L^{\text{MLE}}(\theta)$$

of the following loss function:

$$\begin{aligned} L^{\text{MLE}}(\theta) &= \sum_{s=1}^{S} \log p_\theta(\mathbf{y}^{(s)} | \mathbf{x}^{(s)}) \\ &= \sum_{s=1}^{S} \sum_{t=1}^{T_y} \log p_\theta(y_t | \mathbf{x}^{(s)}, \mathbf{y}^{(s)}_{<t}). \end{aligned}$$

This loss function is non-convex for the case of neural networks. Clever initialization strategies, adaptive learning rates and momentum techniques are required to find good local maxima and to speed up convergence (Sutskever et al., 2013). Another trick of the trade is to ensemble several models with different random initializations to improve over single models (Luong et al., 2015a).

At test time, we face a search problem to find the sequence of target words with the highest probability. Beam search reduces the search error in comparison to greedy search, but also exponentially increases decoding time.

**Algorithm 1** Neural Bandit Structured Prediction

**Input:** Sequence of learning rates $\gamma_k$
**Output:** Optimal parameters $\hat{\theta}$

Initialize $\theta_0$
**for** $k = 0, \ldots, K$ **do**
  Observe $\mathbf{x}_k$
  Sample $\tilde{\mathbf{y}}_k \sim p_\theta(\mathbf{y}|\mathbf{x}_k)$
  Obtain feedback $\Delta(\tilde{\mathbf{y}}_k)$
  $\theta_{k+1} = \theta_k - \gamma_k \, s_k$
Choose a solution $\hat{\theta}$ from the list $\{\theta_0, \ldots, \theta_K\}$

## 4 Neural Bandit Structured Prediction

Algorithm 1 is an adaptation of the Bandit Structured Prediction algorithm of Sokolov et al. (2016b) to neural models: For $K$ rounds, a model with parameters $\theta$ receives an input, samples an output structure, and receives user feedback. Based on this feedback, a stochastic gradient $s_k$ is computed and the model parameters are updated. As a post-optimization step, a solution $\hat{\theta}$ is selected from the iterates. This is done with online-to-batch conversion by choosing the model with optimal performance on held-out data.

The core of the algorithm is the sampling: if the model distribution is very peaked, the model exploits, i.e., it presents the most probable outputs to the user. If the distribution is close to uniform, the model explores, i.e., it presents random outputs to the user. The balance between exploitation and exploration is crucial to the learning process: in the beginning the model is rather uninformed and needs to explore in order to find outputs with high reward, while in the end it ideally converges towards a peaked distribution that exactly fits the user's needs. Pre-training the model, i.e. setting $\theta_0$ wisely, ensures a reasonable exploitation-exploration trade-off.

This online learning algorithm can be applied

to any objective $L$ provided the stochastic gradients $s_k$ are unbiased estimators of the true gradient of the objective, i.e., we require $\nabla L = \mathbb{E}[s_k]$. In the following, we will present objectives from Sokolov et al. (2016b) transferred to neural models, and explain how they can be enhanced by control variates.

### 4.1 Expected Loss (EL) Minimization

The first objective is defined as the expectation of a task loss $\Delta(\tilde{\mathbf{y}})$, e.g. $-\text{BLEU}(\tilde{\mathbf{y}})$, over all input and output structures:

$$L^{\text{EL}}(\theta) = \mathbb{E}_{p(\mathbf{x})\, p_\theta(\tilde{\mathbf{y}}|\mathbf{x})} \left[\Delta(\tilde{\mathbf{y}})\right]. \tag{1}$$

In the case of full-information learning where reference outputs are available, we could evaluate all possible outputs against the reference to obtain an exact estimation of the loss function. However, this is not feasible in our setting since we only receive partial feedback for a single output structure per input. Instead, we use stochastic approximation to optimize this loss. The stochastic gradient for this objective is computed as follows:

$$s_k^{\text{EL}} = \Delta(\tilde{\mathbf{y}}) \frac{\partial \log p_\theta(\tilde{\mathbf{y}}|\mathbf{x}_k)}{\partial \theta}. \tag{2}$$

Objective (1) is known from minimum risk training (Och, 2003) and has been lifted to NMT by Shen et al. (2016) – but not for learning from weak feedback. Equation (2) is an instance of the score function gradient estimator (Fu, 2006) where

$$\nabla \log p_\theta(\tilde{\mathbf{y}}|\mathbf{x}_k) \tag{3}$$

denotes the score function. We give an algorithm to sample structures from an encoder-decoder model in Algorithm 2. It corresponds to the algorithm presented by Shen et al. (2016) with the difference that it samples single structures, does not assume a reference structure, and additionally returns the sample probabilities. A similar objective has also been used in the REINFORCE algorithm (Williams, 1992) which has been adapted to NMT by Ranzato et al. (2016).

### 4.2 Pairwise Preference Ranking (PR)

The previous objective requires numerical feedback as an estimate of translation quality. Alternatively, we can learn from pairwise preference judgments that are formalized in preference ranking objectives. Let $\mathcal{P}(\mathbf{x}) = \{\langle \mathbf{y}_i, \mathbf{y}_j \rangle \,|\mathbf{y}_i, \mathbf{y}_j \in \mathcal{Y}(\mathbf{x})\}$ denote the set of output pairs for an input $\mathbf{x}$, and let $\Delta(\langle \mathbf{y}_i, \mathbf{y}_j \rangle) : \mathcal{P}(\mathbf{x}) \rightarrow [0, 1]$ denote a task loss function that specifies a dispreference of $y_i$ over $y_j$. In our experimental simulations we use two types of pairwise feedback. Firstly, continuous pairwise feedback[2] is computed as

$$\Delta(\langle \mathbf{y}_i, \mathbf{y}_j \rangle) = \Delta(\mathbf{y}_j) - \Delta(\mathbf{y}_i),$$

and secondly, binary feedback is computed as

$$\Delta(\langle \mathbf{y}_i, \mathbf{y}_j \rangle) = \begin{cases} 1 & \text{if } \Delta(\mathbf{y}_j) > \Delta(\mathbf{y}_i), \\ 0 & \text{otherwise.} \end{cases}$$

Analogously to the sequence-level sampling for linear models (Sokolov et al., 2016b), we define the following probabilities for word-level sampling:

$$p_\theta^+(\tilde{y}_t = w_i|\mathbf{x}, \hat{\mathbf{y}}_{<t}) = \frac{\exp(o_{w_i})}{\sum_{v=1}^{V} \exp(o_{w_v})},$$
$$p_\theta^-(\tilde{y}_t = w_j|\mathbf{x}, \hat{\mathbf{y}}_{<t}) = \frac{\exp(-o_{w_j})}{\sum_{v=1}^{V} \exp(-o_{w_v})}.$$

The effect of the negation within the softmax is that the two distributions $p_\theta^+$ and $p_\theta^-$ rank the next candidate target words $\tilde{y}_t$ (given the same history, here the greedy output $\hat{\mathbf{y}}_{<t}$) in opposite order. Globally normalized models as in the linear case, or LSTM-CRFs (Huang et al., 2015) for the non-linear case would allow sampling full structures such that the ranking over full structures is reversed. But in the case of locally normalized RNNs we retrieve only locally reversed-rank samples. Since we want the model to learn to rank $\tilde{\mathbf{y}}_i$ over $\tilde{\mathbf{y}}_j$, we would have to sample $\tilde{\mathbf{y}}_i$ word-by-word from $p_\theta^+$ and $\tilde{\mathbf{y}}_j$ from $p_\theta^-$. However, sampling all words of $\tilde{\mathbf{y}}_j$ from $p_\theta^-$ leads to translations that are neither fluent nor source-related, so we propose to randomly choose one position of $\tilde{\mathbf{y}}_j$ where the next word is sampled from $p_\theta^-$ and sample the remaining words from $p_\theta^+$. We found that this method produces suitable negative samples, which are only slightly perturbed and still relatively fluent and source-related. A detailed algorithm is given in Algorithm 3.

In the same manner as for linear models, we define the probability of a pair of sequences as

$$p_\theta(\langle \tilde{\mathbf{y}}_i, \tilde{\mathbf{y}}_j \rangle |\mathbf{x}) = p_\theta^+(\tilde{\mathbf{y}}_i|\mathbf{x}) \times p_\theta^-(\tilde{\mathbf{y}}_j|\mathbf{x}).$$

[2]Note that our definition of continuous feedback is slightly different from the one proposed in Sokolov et al. (2016b) where updates are only made for misrankings.

**Algorithm 2** Sampling Structures

**Input:** Model $\theta$, target sequence length limit $T_y$
**Output:** Sequence of words $\mathbf{w} = (w_1, \ldots, w_{Ty})$ and log-probability $p$

$w_0 = \text{START}, p_0 = 0$
$\mathbf{w} = (w_0)$
**for** $t \leftarrow 1 \ldots T_y$ **do**
  $w_t \sim p_\theta(w|\mathbf{x}, \mathbf{w}_{<t})$
  $p_t = p_{t-1} + \log p_\theta(w|\mathbf{x}, \mathbf{w}_{<t})$
  $\mathbf{w} = (w_1, \ldots, w_{t-1}, w_t)$
**end for**
Return $\mathbf{w}$ and $p_T$

Note that with the word-based sampling scheme described above, the sequence $\tilde{\mathbf{y}}_j$ also includes words sampled from $p_\theta^+$.
The pairwise preference ranking objective expresses an expectation over losses over these pairs:

$$L^{\text{PR}}(\theta) = \mathbb{E}_{p(\mathbf{x})\, p_\theta(\langle \tilde{\mathbf{y}}_i, \tilde{\mathbf{y}}_j \rangle | \mathbf{x})} \left[ \Delta(\langle \tilde{\mathbf{y}}_i, \tilde{\mathbf{y}}_j \rangle) \right]. \quad (4)$$

The stochastic gradient for this objective is

$$s_k^{\text{PR}} = \Delta(\langle \tilde{\mathbf{y}}_i, \tilde{\mathbf{y}}_j \rangle) \times \left( \frac{\partial \log p_\theta^+(\tilde{\mathbf{y}}_i|\mathbf{x}_k)}{\partial \theta} + \frac{\partial \log p_\theta^-(\tilde{\mathbf{y}}_j|\mathbf{x}_k)}{\partial \theta} \right). \quad (5)$$

This training procedure resembles well-known approaches for noise contrastive estimation (Gutmann and Hyvärinen, 2010) with negative sampling that are commonly used for neural language modeling (Collobert et al., 2011; Mnih and Teh, 2012; Mikolov et al., 2013). In these approaches, negative samples are drawn from a non-parametric noise distribution, whereas we draw them from the perturbed model distribution.

### 4.3 Control Variates

The stochastic gradients defined in equations (2) and (5) can be used in stochastic gradient descent optimization (Bottou et al., 2016) where the full gradient is approximated using a minibatch or a single example in each update. The stochastic choice, in our case on inputs and outputs, introduces noise that leads to slower convergence and degrades performance. In the following, we explain how antithetic and additive control variate techniques from Monte Carlo simulation (Ross, 2013) can be used to remedy these problems.

The idea of additive control variates is to augment a random variable $X$ whose expectation is sought, by another random variable $Y$ to which $X$ is highly correlated. $Y$ is then called the control variate. Let $\bar{Y}$ furthermore denote its expectation. Then the following quantity $X - \hat{c}\,Y + \hat{c}\,\bar{Y}$ is an unbiased estimator of $\mathbb{E}[X]$. In our case, the random variable of interest is the noisy gradient $X = s_k$ from Equation (2). The variance reduction effect of control variates can be seen by computing the variance of this quantity:

$$\text{Var}(X - \hat{c}\,Y) = \text{Var}(X) + \hat{c}^2 \text{Var}(Y) - 2\hat{c}\,\text{Cov}(X, Y). \quad (6)$$

Choosing a control variate such that $\text{Cov}(X, Y)$ is positive and high enough, the variance of the gradient estimate will be reduced.

An example is the average reward baseline known from reinforcement learning (Williams, 1992), yielding

$$Y_k = \nabla \log p_\theta(\tilde{\mathbf{y}}|\mathbf{x}_k) \frac{1}{k} \sum_{j=1}^{k} \Delta(\tilde{\mathbf{y}}_j). \quad (7)$$

The optimal scalar $\hat{c}$ can be derived easily by taking the derivative of (6), leading to $\hat{c} = \frac{\text{Cov}(X,Y)}{\text{Var}(X)}$. This technique has been applied to using the score

**Algorithm 3** Sampling Pairs of Structures

**Input:** Model $\theta$, target sequence length limit $T_y$
**Output:** Pair of sequences $\langle \mathbf{w}, \mathbf{w}' \rangle$ and their log-probability $p$

$p_0 = 0$
$\mathbf{w}, \mathbf{w}', \hat{\mathbf{w}} = (\text{START})$
$i \sim \mathcal{U}(1, T)$
**for** $t \leftarrow 1 \ldots T_y$ **do**
  $\hat{w}_t = \arg\max_{w \in V} p_\theta^+(w|\mathbf{x}, \hat{\mathbf{w}}_{<t})$
  $w_t \sim p_\theta^+(w|\mathbf{x}, \hat{\mathbf{w}}_{<t})$
  $p_t = p_{t-1} + \log p_\theta^+(w_t|\mathbf{x}, \hat{\mathbf{w}}_{<t})$
  **if** $i = t$ **then**
    $w'_t \sim p_\theta^-(w|\mathbf{x}, \hat{\mathbf{w}}_{<t})$
    $p_t = p_t + \log p_\theta^-(w'_t|\mathbf{x}, \hat{\mathbf{w}}_{<t})$
  **else**
    $w'_t \sim p_\theta^+(w|\mathbf{x}, \hat{\mathbf{w}}_{<t})$
    $p_t = p_t + \log p_\theta^+(w'_t|\mathbf{x}, \hat{\mathbf{w}}_{<t})$
  **end if**
  $\mathbf{w} = (w_1, \ldots, w_{t-1}, w_t)$
  $\mathbf{w}' = (w'_1, \ldots, w'_{t-1}, w'_t)$
  $\hat{\mathbf{w}} = (\hat{w}_1, \ldots, \hat{w}_{t-1}, \hat{w}_t)$
**end for**
Return $\langle \mathbf{w}, \mathbf{w}' \rangle$ and $p_T$

function (Equation (3)) as control variate in Ranganath et al. (2014), yielding the following control variate:

$$Y_k = \nabla \log p_\theta(\tilde{\mathbf{y}}|\mathbf{x}_k). \quad (8)$$

Note that for both types of control variates, (7) and (8), the expectation $\bar{Y}$ is zero, simplifying the implementation. However, the optimal scalar $\hat{c}$ has to be estimated for every entry of the gradient separately for the score function control variate. We will explore both types of control variates for the stochastic gradient (2) in our experiments.

A further effect of control variates is to reduce the magnitude of the gradient, the more so the more the stochastic gradient and the control variate covary. For $L$-Lipschitz continuous functions, a reduced gradient norm directly leads to a bound on $L$ which appears in the algorithmic stability bounds of Hardt et al. (2016). This effect of improved generalization by control variates is empirically validated in our experiments.

A similar variance reduction effect can be obtained by antithetic control variates. Here $\mathbb{E}[X]$ is approximated by the estimator $\frac{X_1+X_2}{2}$ whose variance is

$$\mathrm{Var}\left(\frac{X_1+X_2}{2}\right) = \frac{1}{4}\big(\mathrm{Var}(X_1) + \mathrm{Var}(X_2) + 2\mathrm{Cov}(X_1, X_2)\big). \quad (9)$$

Choosing the variates $X_1$ and $X_2$ such that $\mathrm{Cov}(X_1, X_2)$ is negative will reduce the variance of the gradient estimate. Under certain assumptions, the stochastic gradient (5) of the pairwise preference objective can be interpreted as an antithetic estimator of the score function (3). The antithetic variates in this case would be

$$X_1 = \nabla \log p^+_\theta(\tilde{\mathbf{y}}_i|\mathbf{x}_k), \quad (10)$$
$$X_2 = \nabla \log p^-_\theta(\tilde{\mathbf{y}}_j|\mathbf{x}_k),$$

where an antithetic dependence of $X_2$ on $X_1$ can be achieved by construction of $p^+_\theta$ and $p^-_\theta$ (see Capriotti (2008) which is loosely related to our approach). Similar to control variates, antithetic variates have the effect of shrinking the gradient norm, the more so the more the variates are antithetically correlated, leading to possible improvements in algorithmic stability (Hardt et al., 2016).

## 5 Experiments

In the following, we present an experimental evaluation of the learning objectives presented above on machine translation domain adaptation. We compare how the presented neural bandit learning objectives perform in comparison to linear models, then discuss the handling of unknown words and eventually investigate the impact of techniques for variance reduction.

| Domain | Version | Train | Valid. | Test |
|---|---|---|---|---|
| Europarl | v.5 | 1.6M | 2k | 2k |
| News Commentary | WMT07 | 40k | 1k | 2k |
| TED | TED2013 | 153k | 2k | 2k |

Table 1: Number of parallel sentences for training, validation and test sets for French-to-English domain adaptation.

### 5.1 Setup

**Data.** We perform domain adaptation from Europarl (EP) to News Commentary (NC) and TED talks (TED) for translations from French to English. Table 1 provides details about the datasets. For data pre-processing we follow the procedure of Sokolov et al. (2016a,b) using `cdec` tools for filtering, lowercasing and tokenization. The challenge for the bandit learner is to adapt from the EP domain to NC or TED with weak feedback only.

**NMT Models.** We choose a standard encoder-decoder architecture with single-layer GRU RNNs with 800 hidden units, a word embedding size of 300 and `tanh` activations. The encoder consists of a bidirectional RNN, where the hidden states of backward and forward RNN are concatenated. The decoder uses the attention mechanism proposed by Bahdanau et al. (2015).[3] Source and target vocabularies contain the 30k most frequent words of the respective parts of the training corpus. We limit the maximum sentence length to 50. Dropout (Srivastava et al., 2014) with a probability of 0.5 is applied to the network in several places: on the embedded inputs, before the output layer, and on the initial state of the decoder RNN. The gradient is clipped when its norms exceeds 1.0 to prevent exploding gradients and stabilize learning (Pascanu et al., 2013). All models are implemented and trained with the sequence learning framework `Neural Monkey` (Libovický et al.,

[3] We do not use beam search nor ensembling, although we are aware that higher performance is almost guaranteed with these techniques. Our goal is to show relative differences between different models, so a simple setup is sufficient for the purpose of our experiments.

2016; Bojar et al., 2016).[4] They are trained with a minibatch size of 20, fitting onto single 8GB GPU machines. The training dataset is shuffled before each epoch.

**Baselines.** The out-of-domain baseline is trained on the EP training set with standard MLE. For both NC and TED domains, we train two full-information in-domain baselines: The first in-domain baseline is trained on the relatively small in-domain training data. The second in-domain baseline starts from the out-of-domain model and is further trained on the in-domain data. All baselines are trained with MLE and Adam (Kingma and Ba, 2014) ($\alpha = 1 \times 10^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$) until their performance stops increasing on respective held-out validation sets. The gap between the performance of the out-of-domain model and the in-domain models defines the range of possible improvements for bandit learning. All models are evaluated with Neural Monkey's `mteval`. For statistical significance tests we used Approximate Randomization testing (Noreen, 1989).

**Bandit Learning.** Bandit learning starts with the parameters of the out-of-domain baseline. The bandit models are expected to improve over the out-of-domain baseline by receiving feedback from the new domain, but at most to reach the in-domain baseline since the feedback is weak. The models are trained with Adam on in-domain data for at most 20 epochs. Adam's step-size parameter $\alpha$ was tuned on the validation set and was found to perform best when set to $1 \times 10^{-5}$ for non-pairwise, $1 \times 10^{-6}$ for pairwise objectives on NC, $1 \times 10^{-7}$ for pairwise objectives on TED. The best model parameters, selected with early stopping on the in-domain validation set, are evaluated on the held-out in-domain test set. In the spirit of Freitag and Al-Onaizan (2016) they are additionally evaluated on the out-of-domain test set to investigate how much knowledge of the old domain the models lose while adapting to the new domain. Bandit learning experiments are repeated two times, with different random seeds, and mean BLEU scores with standard deviation are reported.

[4]The `Neural Monkey` fork `https://github.com/juliakreutzer/bandit-neuralmonkey` contains bandit learning objectives and the configuration files for our experiments.

**Feedback Simulation.** Weak feedback is simulated from the target side of the parallel corpus, but references are never revealed to the learner. Sokolov et al. (2016a,b) used a smoothed version of per-sentence BLEU for simulating the weak feedback for generated translations from the comparison with reference translations. Here, we use gGLEU instead, which Wu et al. (2016) recently introduced for learning from sentence-level reward signals correlating well with corpus BLEU. This metric is closely related to BLEU, but does not have a brevity penalty and considers the recall of matching $n$-grams. It is defined as the minimum of recall and precision over the total $n$-grams up to a certain $n$. Hence, for our experiments $\Delta(\tilde{\mathbf{y}}) = -\text{gGLEU}(\tilde{\mathbf{y}}, \mathbf{y})$, where $\tilde{\mathbf{y}}$ is a sample translation and $\mathbf{y}$ is the reference translation.

**Unknown words.** One drawback of NMT models is their limitation to a fixed source- and target vocabulary. In a domain adaptation setting, this limitation has a critical impact to the translation quality. The larger the distance between old and new domain, the more words in the new domain are unknown to the models trained on the old domain (represented with a special UNK token). We consider two strategies for this problem for our experiments:

1. UNK-Replace: Jean et al. (2015) and Luong et al. (2015b) replace generated UNK tokens with aligned source words or their lexical translations in a post-processing step. Freitag and Al-Onaizan (2016) and Hashimoto et al. (2016) demonstrated that this technique is beneficial for NMT domain adaptation.

2. BPE: Sennrich et al. (2016) introduce byte pair encoding (BPE) for word segmentation to build translation models on sub-word units. Rare words are decomposed into sub-word units, while the most frequent words remain single vocabulary items.

For UNK-Replace we use `fast_align` to generate lexical translations on the EP training data. When an UNK token is generated, we look up the attention weights and find the source token that receives most attention in this step. If possible, we replace the UNK token by its lexical translation. If it is not included in the lexical translations, it is replaced by the source token. The main benefit of this technique is that it deals well

| Algorithm | Train data | Iter. | EP | NC | TED |
|---|---|---|---|---|---|
| MLE | EP | 12.3M | 31.44 | 26.98 | 23.48 |
| MLE-UNK | | | 31.82 | 28.00 | 24.59 |
| MLE-BPE | | 12.0M | 31.81 | 27.20 | 24.35 |

Table 2: Out-of-domain NMT baseline results (BLEU) on in- and out-of-domain test sets trained only on EP data.

with unknown named entities that are just passed through from source to target. However, since it is a non-differentiable post-processing step, the NMT model cannot directly be trained for this behavior. Therefore we also train sub-word level NMT with BPE. We apply 29,800 merge operations to obtain a vocabulary of 29,908 sub-words. The procedure for training these models is exactly the same as for the word-based models. The advantage of this method is that the model is in principle able to generate any word composing it from sub-word units. However, training sequences become longer and candidate translations are sampled on a sub-word level, which introduces the risk of sampling nonsense words.

**Control variates.** We implement the average baseline control variate as defined in Equation 7, which results in keeping an running average over previous losses. Intuitively, absolute gGLEU feedback is turned into relative feedback that reflects the current state of the model. The sign of the update is switched when the gGLEU for the current sample is worse than the average gGLEU, so the model makes a step away from it, while in the case of absolute feedback it would still make a small step towards it. In addition, we implement the score function control variate with a running estimate $\hat{c}_k = \frac{1}{k}\sum_{j=1}^{k} \frac{\mathrm{Cov}(s_j, \nabla \log p_\theta(\tilde{\mathbf{y}}_j|\mathbf{x}_j))}{\mathrm{Var}(s_j)}$.

### 5.2 Results

In the following, we discuss the results of the experimental evaluation of the models described above. The out-of-domain baseline results are given in Table 2, those for the in-domain baselines in 3. The results for bandit learning on NC and TED are reported in Table 4. For bandit learning we give mean improvements over the respective out-of-domain baselines in the Diff.-columns.

**Baselines.** The NMT out-of-domain baselines, reported in Table 2, perform comparable to the linear baseline from Sokolov et al. (2016a,b) on

| Algorithm | Train data | Iter. | EP | NC |
|---|---|---|---|---|
| MLE | NC | 978k | 13.67 | 22.32 |
| MLE-UNK | | | 13.83 | 22.56 |
| MLE-BPE | | 1.0M | 14.09 | 23.01 |
| MLE | EP→NC | 160k | 26.66 | 31.91 |
| MLE-UNK | | | 27.19 | 33.19 |
| MLE-BPE | | 160k | 27.14 | 33.31 |
| **Algorithm** | **Train data** | **Iter.** | **EP** | **TED** |
| MLE | TED | 2.2M | 14.16 | 32.71 |
| MLE-UNK | | | 15.15 | 33.16 |
| MLE-BPE | | 3.0M | 14.18 | 32.81 |
| MLE | EP→TED | 460k | 23.88 | 33.65 |
| MLE-UNK | | | 24.64 | 35.57 |
| MLE-BPE | | 2.2M | 23.39 | 36.23 |

Table 3: In-domain NMT baselines results (BLEU) on in- and out-of-domain test sets. The EP→NC is first trained on EP, then fine-tuned on NC. The EP→TED is first trained on EP, then fine-tuned on TED.

NC, but the in-domain EP→NC (Table 3) baselines outperform the linear baseline by more than 3 BLEU points. Continuing training of a pre-trained out-of-domain model on a small amount of in domain data is very hence effective, whilst the performance of the models solely trained on small in-domain data is highly dependent on the size of this training data set. For TED, the in-domain dataset is almost four times as big as the NC training set, so the in-domain baselines perform better. This effect was previously observed by Luong and Manning (2015) and Freitag and Al-Onaizan (2016).

**Bandit Learning.** The NMT bandit models that optimize the EL objective yield generally a much higher improvement over the out-of-domain models than the corresponding linear models: As listed in Table 4, we find improvements of between 2.33 and 2.89 BLEU points on the NC domain, and between 4.18 and 5.18 BLEU points on the TED domain. In contrast, the linear models with sparse features and hypergraph re-decoding achieved a maximum improvement of 0.82 BLEU points on NC.

Optimization of the PR objective shows improvements of up to 1.79 BLEU points on NC (compared to 0.6 BLEU points for linear models), but no significant improvement on TED. The biggest impact of this variance reduction tech-

| Algorithm | Iter. | EP | NC | Diff. |
|---|---|---|---|---|
| EL | 317k | $30.36_{\pm 0.20}$ | $29.34_{\pm 0.29}$ | 2.36 |
| EL-UNK* | 317k | $30.73_{\pm 0.20}$ | $30.33_{\pm 0.42}$ | 2.33 |
| EL-UNK** | 349k | $30.67_{\pm 0.04}$ | $30.45_{\pm 0.27}$ | 2.45 |
| EL-BPE | 543k | $30.09_{\pm 0.31}$ | $30.09_{\pm 0.01}$ | 2.89 |
| PR-UNK** (bin) | 22k | $30.76_{\pm 0.03}$ | $29.40_{\pm 0.02}$ | 1.40 |
| PR-BPE (bin) | 14k | $31.02_{\pm 0.09}$ | $28.92_{\pm 0.03}$ | 1.72 |
| PR-UNK** (cont) | 12k | $30.81_{\pm 0.02}$ | $29.43_{\pm 0.02}$ | 1.43 |
| PR-BPE (cont) | 8k | $30.91_{\pm 0.01}$ | $28.99_{\pm 0.00}$ | 1.79 |
| SF-EL-UNK** | 713k | $29.97_{\pm 0.09}$ | $30.61_{\pm 0.05}$ | 2.61 |
| SF-EL-BPE | 375k | $30.46_{\pm 0.10}$ | $30.20_{\pm 0.11}$ | 3.00 |
| BL-EL-UNK** | 531k | $30.19_{\pm 0.37}$ | $31.47_{\pm 0.09}$ | 3.47 |
| BL-EL-BPE | 755k | $29.88_{\pm 0.07}$ | $31.28_{\pm 0.24}$ | **4.08** |

(a) Domain adaptation from EP to NC.

| Algorithm | Iter. | EP | TED | Diff. |
|---|---|---|---|---|
| EL | 976k | $29.34_{\pm 0.42}$ | $27.66_{\pm 0.03}$ | 4.18 |
| EL-UNK* | 976k | $29.68_{\pm 0.29}$ | $29.44_{\pm 0.06}$ | 4.85 |
| EL-UNK** | 1.1M | $29.62_{\pm 0.15}$ | $29.77_{\pm 0.15}$ | 5.18 |
| EL-BPE | 831k | $30.03_{\pm 0.43}$ | $28.54_{\pm 0.04}$ | 4.18 |
| PR-UNK** (bin) | 14k | $31.84_{\pm 0.01}$ | $24.85_{\pm 0.00}$ | 0.26 |
| PR-BPE (bin) | 69k | $31.77_{\pm 0.01}$ | $24.55_{\pm 0.01}$ | 0.20 |
| PR-UNK** (cont) | 9k | $31.85_{\pm 0.02}$ | $24.85_{\pm 0.01}$ | 0.26 |
| PR-BPE (cont) | 55k | $31.79_{\pm 0.02}$ | $24.59_{\pm 0.01}$ | 0.24 |
| SF-EL-UNK** | 658k | $30.18_{\pm 0.15}$ | $29.12_{\pm 0.10}$ | 4.53 |
| SF-EL-BPE | 590k | $30.32_{\pm 0.26}$ | $28.51_{\pm 0.18}$ | 4.16 |
| BL-EL-UNK** | 644k | $29.91_{\pm 0.03}$ | $30.44_{\pm 0.13}$ | 5.85 |
| BL-EL-BPE | 742k | $29.84_{\pm 0.61}$ | $30.24_{\pm 0.46}$ | **5.89** |

(b) Domain adaptation from EP to TED.

Table 4: Bandit NMT results (BLEU) on EP, NC and TED test sets. UNK* models involve UNK replacement only during testing, UNK** include UNK replacement already during training. For PR, either binary (bin) or continuous feedback (cont) was used. Control variates: average reward baseline (BL) and score function (SF). Results are averaged over two independent runs and standard deviation is given in subscripts. Improvements over respective out-of-domain models are given in the Diff.-columns.

nique is a considerable speedup of training speed of 1 to 2 orders of magnitude compared to EL.

A beneficial side-effect of NMT learning from weak feedback is that the knowledge from the out-domain training is not simply "overwritten". This happens to full-information in-domain tuning where more than 4 BLEU points are lost in an evaluation on the out-domain data. On the contrary, the bandit learning models still achieve high results on the original domain. This is useful for conservative domain adaptation, where the performance of the models in the old domain is still relevant.

**Unknown words.** By handling unknown words with UNK-Replace or BPEs, we find consistent improvements over the plain word-based models for all baselines and bandit learning models. We observe that the models with UNK replacement essentially benefit from passing through source tokens, and only marginally from lexical translations. Bandit learning models take particular advantage of UNK replacement when it is included already during training. The sub-word models achieve the overall highest improvement over the baselines, although sometimes generating nonsense words.

**Control variates.** Applying the score function control variate to EL optimization does not largely change learning speed or BLEU results. However, the average reward control variate leads to improvements of around 1 BLEU over the EL optimization without variance reduction on both domains.

## 6 Conclusion

In this paper, we showed how to lift structured prediction under bandit feedback from linear models to non-linear sequence-to-sequence learning using recurrent neural networks with attention. We introduced algorithms to train these models under numerical feedback to single output structures or under preference rankings over pairs of structures. In our experimental evaluation on the task of neural machine translation domain adaptation, we found relative improvements of up to 5.89 BLEU points over out-of-domain seed models, outperforming also linear bandit models. Furthermore, we argued that pairwise ranking under bandit feedback can be interpreted as a use of antithetic variates, and we showed how to include average reward and score function baselines as control variates for improved training speed and generalization. In future work, we would like to apply the presented non-linear bandit learners to other structured prediction tasks.

## Acknowledgments

This research was supported in part by the German research foundation (DFG), and in part by a research cooperation grant with the Amazon Development Center Germany.

## References


Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*. San Diego, CA.

Ondřej Bojar, Roman Sudarikov, Tom Kocmi, Jindřich Helcl, and Ondřej Cıfka. 2016. UFAL submissions to the IWSLT 2016 MT track. In *IWSLT*. Seattle, WA.

Leon Bottou, Frank E. Curtis, and Jorge Nocedal. 2016. Optimization methods for large-scale machine learning. *eprint arXiv:1606.04838v1* .

Luca Capriotti. 2008. Reducing the variance of likelihood ratio greeks in Monte Carlo. In *WCS*. Miami, FL.

Kyunghyun Cho, Bart van Merriënboer, Çalar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*. Doha, Qatar.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *eprint arXiv:1412.3555* .

James Clarke, Dan Goldwasser, Wing-Wei Chang, and Dan Roth. 2010. Driving semantic parsing from the world's response. In *CoNLL*. Portland, OR.

Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *JMLR* 12:2461–2505.

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

Markus Freitag and Yaser Al-Onaizan. 2016. Fast domain adaptation for neural machine translation. *eprint arXiv:1612.06897* .

Michael C. Fu. 2006. Gradient estimation. In S.G. Henderson and B.L. Nelson, editors, *Handbook in Operations Research and Management Science*, volume 13, pages 575–616.

Kevin Gimpel and Noah A. Smith. 2010. Softmax-margin training for structured log-linear models. Technical Report CMU-LTI-10-008, Carnegie Mellon University.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*. Sardinia, Italy.

Moritz Hardt, Ben Recht, and Yoram Singer. 2016. Train faster, generalize better: Stability of stochastic gradient descent. In *ICML*. New York, NY.

Kazuma Hashimoto, Akiko Eriguchi, and Yoshimasa Tsuruoka. 2016. Domain adaptation and attention-based unknown word replacement in chinese-to-japanese neural machine translation. In *COLING Workshop on Asian Translation*. Osaka, Japan.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tieyan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *NIPS*. Barcelona, Spain.

Xiaodong He and Li Deng. 2012. Maximum expected BLEU training of phrase and lexicon translation models. In *ACL*. Jeju Island, Korea.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *eprint arXiv:1508.01991* .

Sébastien Jean, Orhan Firat, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. Montreal neural machine translation systems for WMT'15. In *WMT*. Lisbon, Portugal.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. In *ACL*. Berlin, Germany.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *eprint arXiv:1412.6980* .

Percy Liang, Michael I. Jordan, and Dan Klein. 2011. Learning dependency-based compositional semantics. In *ACL-HLT*. Portland, OR.

Jindřich Libovickỳ, Jindřich Helcl, Marek Tlustỳ, Pavel Pecina, and Ondřej Bojar. 2016. CUNI system for WMT16 automatic post-editing and multimodal translation tasks. In *WMT*. Berlin, Germany.

Minh-Thang Luong and Christopher D. Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *IWSLT*. Da Nang, Vietnam.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015a. Effective approaches to attention-based neural machine translation. In *EMNLP*. Lisbon, Portugal.

Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals, and Wojciech Zaremba. 2015b. Addressing the rare word problem in neural machine translation. In *ACL*. Beijing, China.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*. Lake Tahoe, CA.

Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. In *ICML*. Edinburgh, Scotland.

Eric W. Noreen. 1989. *Computer Intensive Methods for Testing Hypotheses. An Introduction*. Wiley.

Franz J. Och. 2003. Minimum error rate training in statistical machine translation. In *HLT-NAACL*. Edmonton, Canada.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *ICML*. Atlanta, GA.

Rajesh Ranganath, Sean Gerrish, and David M. Blei. 2014. Black box variational inference. In *AISTATS*. Reykjavik, Iceland.

MarcAurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *ICLR*. San Juan, Puerto Rico.

Sheldon M. Ross. 2013. *Simulation*. Elsevier, fifth edition.

Stéphane Ross, Geoffrey J Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*. Ft. Lauderdale, FL.

Stefan Schaal. 1999. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences* 3(6):233–242.

John Schulman, Nicolas Heess, Theophane Weber, and Pieter Abbeel. 2015. Gradient estimation using stochastic computation graphs. In *NIPS*. Montreal, Canada.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *ACL*. Berlin, Germany.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Minimum risk training for neural machine translation. In *ACL*. Berlin, Germany.

David A. Smith and Jason Eisner. 2006. Minimum risk annealing for training log-linear models. In *COLING-ACL*. Sydney, Australia.

Artem Sokolov, Julia Kreutzer, Christopher Lo, and Stefan Riezler. 2016a. Learning structured predictors from bandit feedback for interactive NLP. In *ACL*. Berlin, Germany.

Artem Sokolov, Julia Kreutzer, Christopher Lo, and Stefan Riezler. 2016b. Stochastic structured prediction under bandit feedback. In *NIPS*. Barcelona, Spain.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR* 15(1):1929–1958.

Ilya Sutskever, James Martens, George E. Dahl, and Geoffrey E. Hinton. 2013. On the importance of initialization and momentum in deep learning. In *ICML*. Atlanta, GA.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*. Montreal, Canada.

Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement Learning. An Introduction*. The MIT Press.

Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*. Vancouver, Canada.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *NIPS*. Montreal, Canada.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 20:229–256.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *eprint arXiv:1609.08144* .

Alan Yuille and Xuming He. 2012. Probabilistic models of vision and max-margin methods. *Frontiers of Electrical and Electronic Engineering* 7(1):94–106.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *UAI*. Edinburgh, Scotland.