INDUCING GRAMMARS WITH AND FOR NEURAL MACHINE TRANSLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Previous work has demonstrated the benefits of incorporating additional linguistic annotations such as syntactic trees into neural machine translation. However the cost of obtaining those syntactic annotations is expensive for many languages and the quality of unsupervised learning linguistic structures is too poor to be helpful. In this work, we aim to improve neural machine translation via source side dependency syntax but without explicit annotation. We propose a set of models that learn to induce dependency trees on the source side and learn to use that information on the target side. Importantly, we also show that our dependency trees capture important syntactic features of language and improve translation quality on two language pairs En-De and En-Ru.

1 INTRODUCTION

Sequence to sequence (seq2seq) models have exploded in popularity due to their apparent simplicity and yet surprising modeling strength. The basic architecture cleanly extends the standard machine learning paradigm wherein some function $f$ is learned to map inputs to outputs $x \rightarrow y$ to the case where $x$ and $y$ are natural language strings. In its most basic form, an input is summarized by a recurrent neural network into a summary vector and then decoded into a sequence of observations. These models have been strengthened with optimization techniques like Adam (Kingma & Ba, 2014), Attention (Bahdanau et al., 2015), RNN Dropout (Gal & Ghahramani, 2016), in addition to important advances in expressivity via gating like Long Short-Term Memory (LSTM) cells (Hochreiter & Schmidhuber, 1997).

Despite these impressive advances, the community has still largely been at a loss to explain how these models are so successful at a wide range of linguistic tasks. Recent work has shown that the LSTM-RNN captures a surprising amount of syntax (Linzen et al., 2016), but this is evaluated via downstream tasks designed to test the model’s abilities not its representation.

Simultaneously, recent research in neural machine translation (NMT) has shown the benefit of modeling syntax explicitly using parse trees (Bastings et al., 2017; Li et al., 2017; Eriguchi et al., 2017) rather than assuming the model will discover it. Li et al. (2017) present a mixed encoding of words and a linearized constituency-based parse tree of the source sentence. Bastings et al. (2017) propose to use Graph Convolution to encoder source sentences given their dependency links and attachment labels. In this work, we attempt to contribute to both modeling syntax and investigating a more interpretable interface for testing the syntactic content of a new Seq2Seq model’s internal representation and attention.

We achieve this by augmenting seq2seq with a gate that allows the model to decide between syntactic and semantic objectives. The syntactic objective is encoded via a syntactic structured attention (Section 3) from which we can extract dependency trees. Our goal is to have a model which reaps the benefits of syntactic information (i.e. parse trees) without requiring explicit annotation. In this way, learning the internal representation of our model is a cousin to work done in unsupervised grammar induction except that by focusing on translation we require both syntactic and semantic knowledge.
To motive our work and the importance of structure in translation, consider the process of translating the sentence “The boy sitting next to the girls orders a Miso ramen.” from English to German. The dependency tree of the sentence is given in figure 1. In German, translating the verb “orders”, requires knowledge of its subject “boy” to correctly predict the verb’s gender. This is a case where syntactic agreement requires long-distance information transfer. On the other hand, translating the word “next” can be done in isolation without knowledge of neither its head nor child dependencies. While its true the decoder can, in principle, utilize previously predicted words (e.g. the translation of the “boy”) to reason about subject-verb agreement, in practice LSTMs still struggle with long-distance dependencies. Moreover, Belinkov et al. (2017) showed that using attention reduces the capacity of the decoder to learn target side syntax.

Based on the insights we see from examples that the one above, we have designed a model with the following properties:

1. It can discover syntactic relations in the source sentences;
2. It can decide the amount of syntactic information from the source to use when generating target words.

Previous work seems to imply that syntactic dependencies on the source side can be modeled via a self-attention layer (Vaswani et al., 2017) because self-attention allows direct interactions amongst source words. However, we will show that this is not in fact the case (section §6). We achieve our first requirement (1) by means of a syntactic attention layer (§3.1) that imposes non-projective dependency structure over the source sentence. To meet our second requirement (2) we use a simple gating mechanism (§3.2) that learns when to use the source side syntax.

As noted previously, in addition to demonstrating improvements in translation quality with our proposed models, we are also interested in analyzing the aforementioned non-projective dependency trees learned by the models. Recent work has begun analyzing task-specific latent trees (Williams et al., 2017). It has been shown that cooperating hierarchical structures leads to better task performance. Unlike the previous work that induce latent trees explicitly for semantic tasks, we present the first results on learning latent trees with a joint syntactic-semantic objective. We do this in the service of machine translation which inherently requires access to both facets of a sentence.

In summary, in this work we make the following contributions:

- We propose a new NMT model that learns the latent structure of the encoder and how to use it during decoding. Our model is language independent and straightforward to apply with Byte-Pair Encoding (BPE) inputs. We show that our model obtains a significant improvement 0.6 BLEU (German→English) and 0.8 BLEU (English→German) over a strong baseline.
- We perform an in-depth analysis of the learned structures on the source side and investigate where the target decoder decides syntax is required.

The rest of the paper is organized as follow: We describe our NMT baseline in section §2. Our proposed models are detailed in section §3. We present the experimental setups and translation results in section §4. In section §5 we analyze models’ behavior by means of visualization which pairs with our analysis of the latent trees induced by our model in section §6. The final section is dedicated to conclusions.
2 NEURAL MACHINE TRANSLATION

Given a training pair of source and target sentences \((x, y)\) of length \(n\) and \(m\) respectively, Neural Machine Translation (NMT) is a conditional probabilistic model \(p(y \mid x)\) implemented using neural networks

\[
\log p(y \mid x; \theta) = \sum_{j=1}^{m} \log p(y_j \mid y_{i<j}, x; \theta)
\]

where \(\theta\) is the model’s parameters. We will omit the parameters \(\theta\) herein for readability.

The NMT system used in this work is a seq2seq model that consists of a bidirectional LSTM encoder and an LSTM decoder coupled with an attention mechanism (Bahdanau et al., 2015; Luong et al., 2015). Our system is based on a PyTorch implementation\(^1\) of OpenNMT (Klein et al., 2017). Let \(\{s_i \in \mathbb{R}^d\}_{i=1}^n\) be the output of the encoder

\[
S = \text{enc}(x)
\]

Here we use \(S = [s_1; \ldots; s_n] \in \mathbb{R}^{d \times n}\) as a concatenation of \(\{s_i\}\). The decoder is composed of stacked LSTMs with input-feeding. Specifically, the inputs of the decoder at time step \(t\) are the previous hidden state \(h_{t-1}\), a concatenation of the embedding of previous generated word \(y_{t-1}\) and a vector \(u_{t-1}\):

\[
u_{t-1} = g(h_{t-1}, c_{t-1})
\]

where \(g\) is a one layer feed-forward network and \(c_{t-1}\) is a context vector computed by an attention mechanism

\[
\alpha = \text{softmax}(h_{t-1}^T W_a S)
\]

\[
c_{t-1} = S \alpha^T
\]

where \(W_a \in \mathbb{R}^{d \times d}\) is a trainable parameter.

Finally a single layer feed-forward network \(f\) takes \(u_t\) as input and returns a multinomial distribution over all the target words

\[
y_t \sim f(u_t)
\]

3 SYNTACTIC ATTENTION MODELS

Previous work on incorporating source-side syntax in NMT often focuses on modifying the standard recurrent encoder such that the encoder is explicitly made aware of the syntactic structure of the source sentence. Given a sentence of length \(n\), syntax encoders of this type return a set of \(n\) annotation vectors each compressing semantic and syntactic relations defined by the given parse tree of the input. The attention module then accesses these annotations during the generation of the target as in figure 2a. We argue that this approach puts a lot of burden on the encoder as it has to balance the influence of semantics and syntax at every step regardless of the target words that are being generated. Here, we propose a simple alternative approach where we let the encoder output two sets of vectors: content vectors and syntactic vectors. The content vectors are the outputs of a standard BiLSTM while the syntactic vectors are produced by a structured attention layer §3.1. Having two separate sets of vectors allows the decoder to decide how much syntax it needs for making a prediction given the decoder’s current state. To implement this idea, we use a standard attention layer to select syntactic context followed by a gating mechanism to control syntactic information. Apart from lifting the burden otherwise placed on the encoder and tightly coupling the syntactic encoding to the decoder, the gating mechanism also allows us to inspect the decoder state and answer the question "When does source side syntax matter?" in section §5.

Inspired by structured attention networks (Kim et al., 2017), we present a syntactic attention layer that aims to discovery and convey source side dependency information to the decoder. The syntactic attention model consists of two parts:

1. A syntactic attention layer for head word selection in the encoder;
2. An attention with gating mechanism to control the amount of syntax needed for generating a target word at each time step.

\(^1\)http://opennmt.net/OpenNMT-py/
3.1 HEAD WORD SELECTION

The head word selection (HS) layer learns to select a soft head word for each source word via structured attention. This layer does not have access to any dependency labels from the source. The HS layer transforms $S$ into a matrix $M$ that encodes implicit dependency structure of $x$ using self-structured-attention. First we apply three trainable weight matrices $W_q, W_k, W_v \in \mathbb{R}^{d \times d}$ to map $S$ to query, key, and value matrices $S_q, S_k, S_v$:

$$
S_q = W_q S \\
S_k = W_k S \\
S_v = W_v S
$$

Then we compute structured attention probabilities relying on a function $s_{attn}$ that we will describe in detail shortly.

$$
\beta = s_{attn}(S_q^T S_k) \\
M = S_v \beta
$$

The structured attention function $s_{attn}$ is inspired by the work of Kim et al. (2017). Let $z \in \{0, 1\}^{n \times n}$ be an adjacency matrix encoding a source’s dependency tree. Let $\phi \in \mathbb{R}^{n \times n}$ be a scoring matrix such that cell $\phi_{i,j}$ scores how likely word $x_i$ is to be the head of word $x_j$. The matrix $\phi$ is obtained simply by

$$
\phi = S_q^T S_k
$$

The probability of a dependency tree $z$ is therefore given by

$$
p(z | x; \phi) = \frac{\exp \left( \sum_{i,j} z_{i,j} \phi_{i,j} \right)}{Z(\phi)}
$$

where $Z(\phi)$ is the partition function.

In the head selection model, we are interested in the marginal $p(z_{i,j} = 1 | x; \phi)$

$$
\beta_{i,j} = p(z_{i,j} = 1 | x; \phi) = \sum_{z : z_{i,j} = 1} p(z | x; \phi)
$$

We use the framework presented by Koo et al. (2007) to compute the marginal of non-projective dependency structures. Koo et al. (2007) use the Kirchhoff’s Matrix-Tree Theorem (Tutte, 1984) to compute $p(z_{i,j} = 1 | x; \phi)$ as follow:

$$
L_{i,j}(\phi) = \begin{cases} 
\sum_{k=1}^{n} \exp(\phi_{k,j}) & \text{if } i = j \\
\exp(\phi_{i,j}) & \text{otherwise}
\end{cases}
$$

Now we construct a matrix $\tilde{L}$ that accounts for root selection

$$
\tilde{L}_{i,j}(\phi) = \begin{cases} 
\exp(\phi_{j,j}) & \text{if } i = 1 \\
L_{i,j}(\phi) & \text{if } i > 1
\end{cases}
$$
The marginals $\beta$ are then
\[
\beta_{i,j} = (1 - \delta_{i,j}) \exp(\phi_{i,j}) \left[ L^{-1}(\phi) \right]_{j,j} - (1 - \delta_{i,1}) \exp(\phi_{i,j}) \left[ L^{-1}(\phi) \right]_{j,i} \tag{14}
\]
where $\delta_{i,j}$ is the Kronecker delta. For the root node, the marginals are given by
\[
\beta_{k,k} = \exp(\phi_{k,k}) \left[ L^{-1}(\phi) \right]_{k,1} \tag{15}
\]

The computation of the marginals is fully differentiable, thus we can train the model in an end-to-end fashion by maximizing the conditional likelihood of the translation. Recently Liu & Lapata (2017) propose using the Matrix-Tree Theorem for evaluating the marginals in end to end training of neural networks. Their work however focuses on semantic objectives rather than a joint semantic and syntactic objectives such as machine translation.

3.2 Incorporating syntactic context

Given the syntactic annotation $M$ (eq. 8), it is straightforward to use this information in the decoder by applying a standard attention mechanism
\[
\gamma = \text{softmax}(h_{t-1}^T W_m M) \tag{16}
\]
\[
d = M \gamma^T \tag{17}
\]
While this is an intuitive approach to constructing a syntactic context vector $d$, there is a little guarantee that the attention will bias toward a syntactic head instead of a semantic one. Without the application of any additional constraints, the attention mechanism is free to attend wherever it prefers. Figure 3 illustrates this problem when the model predicts the translation of the word “orders”. While the attention $\alpha$ is sharp at the content vector for “orders” (in blue), the attention weights $\gamma$ (figure 3a) can also be sharp at the positions where “orders” is the head selected by structured attention (purple). In contrast, when sharing attention weights from the decoder with the encoder (figure 3b), the model is biased to look more at the subject “boy” of the verb “order” via the syntactic annotation $M$, thus helping the decoder choose the correct gender in the translation.

\[
d = M \alpha^T \tag{18}
\]
where $\alpha$ is the attention weight computed in equation 3. We can interpret equation 18 as follow: First the attention will look for the source word $x_j$ to translate, then it will look at the head word $x_i$ of $x_j$ using the same attention weight.

![Figure 3: A pictorial illustration of having two separate attention (3a) and shared attention (3b) from the decoder to the encoder. The blue text represents the content vectors of the sentence and the purple text represents the syntactic vectors. The number corresponding to each word is the probability mass from decoder-to-encoder attention layer(s).](image)

It is not always useful or necessary to access the syntactic context $d$ every time step. Ideally, we should let the model decide whether it needs to use this information. For example, the model might decide when it needs to resolve long distance dependencies in the source side. To control the amount of source side syntactic information we introduce a gating mechanism:
\[
\hat{d}_{t-1} = d_{t-1} \odot \sigma(W_g h_{t-1}) \tag{19}
\]
The vector $h_{t-1}$ from equation 2 now becomes
\[
u_{t-1} = g(h_{t-1}, c_{t-1}, \hat{d}_{t-1}) \tag{20}
\]
3.3 HARD ATTENTION OVER TREE STRUCTURES

We further bias the models to capture syntactic structures by using hard attention. Recall the marginal $\beta_{i,j}$ gives us the probability that word $x_i$ is the head of word $x_j$. We convert these soft weights to hard ones $\bar{\beta}$ by

$$\bar{\beta}_{k,j} = \begin{cases} 1 & \text{if } k = \text{arg max}_i \beta_{i,j} \\ 0 & \text{otherwise} \end{cases} \quad (21)$$

We train this model using the straight-through estimator (Bengio et al., 2013). Note that in this setup, each word has a parent but there is no guarantee that the structure given by hard attention will result in a tree (i.e. it may contain cycle). A more principled way to enforce tree structure is to decode the best tree $T$ using the maximum spanning tree algorithm (Chu & Liu, 1965; Edmonds, 1967) and to set $\bar{\beta}_{k,j} = 1$ if the edge $(x_k \rightarrow x_j) \in T$. Unfortunately, maximum spanning tree decoding can be prohibitively slow as the Chu-Liu-Edmonds algorithm is not GPU friendly. We therefore resort to greedily picking a parent word for each word $x_j$ in the sentence using equation 21. This is actually a principled simplification as greedily assigning a parent for each word is the first step in Chu-Liu-Edmonds algorithm.

4 EXPERIMENTS

Next we will discuss our experimental setup and report results for English+German (En+De) and English+Russian (Ru+En) translation models.

4.1 DATA

We use WMT17 data in our experiments. Table 1 shows the statistics of the data. For En–De, we use a concatenation of Europarl, Common Crawl, Rapid corpus of EU press releases, and News Commentary v12. We use newstest2015 for development and newstest2016, newstest2017 for test. For En–Ru, we use Common Crawl, News Commentary v12, and Yandex Corpus. The development data comes from newstest2016 and newstest2017 and is reserved for testing.

<table>
<thead>
<tr>
<th></th>
<th>Train</th>
<th>Valid</th>
<th>Test</th>
<th>Vocabulary</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>wm16</td>
<td>wm17</td>
<td></td>
<td>En</td>
</tr>
<tr>
<td>En–De</td>
<td>5.9M</td>
<td>2,169</td>
<td>2,999</td>
<td>3,004</td>
</tr>
<tr>
<td>En–Ru</td>
<td>2.1M</td>
<td>2,998</td>
<td>–</td>
<td>3,001</td>
</tr>
</tbody>
</table>

We use BPE (Sennrich et al., 2016) with 32,000 merge operations. We run BPE for each language instead of using BPE for the concatenation of both source and target languages.

4.2 BASELINES

Our baseline is an NMT model with input-feeding (§2). As we will be making several modifications from the basic architecture in our proposed models, we will verify each choice in our architecture design empirically. First we validate the structured attention module by comparing it to a self-attention module (Lin et al., 2017; Vaswani et al., 2017). Since self-attention does not assume any hierarchical structure over the source sentence, we refer it as flat-attention (FA). Second, we validate the benefit of using two sets of annotations in the encoder. We combine the hidden states of the encoder $h$ with syntactic context $d$ to obtain a single set of annotation using the following equation

$$\bar{s}_i = s_i + \sigma(W_g s_i) \odot d_i \quad (22)$$

Here we first down weight the syntactic context before adding it to $s_i$. We refer to this baseline as SA-NMT-1set. Note that in this baseline, there is only one attention layer from the target to the source.
In all the models, we share the weights of target word embeddings and the output layer as suggested by Inan et al. (2016); Press & Wolf (2017).

4.3 Hyper-parameters and Training

For all the models, we set the word embedding size to 1024, the number of LSTM layers to 2, and the dropout rate to 0.3. Parameters are initialized uniformly in \((-0.04, 0.04)\). We use the Adam optimizer with an initial learning rate 0.001. We evaluate our models on development data every 10,000 updates for De-En and 5,000 updates for Ru-En. If the validation perplexity increases, we decay the learning rate by 0.5. We stop training after decaying the learning rate five times as suggested by Denkowski & Neubig (2017). The mini-batch size is 32 in all the experiments. We report the BLEU scores using the multi-bleu.perl script.

4.4 Results

Table 2 shows the BLEU scores in our experiments. We test statistical significance using bootstrap resampling Riezler & Maxwell (2005). Statistical significance are marked as $p < 0.05$ and $p < 0.01$ when compared against the baselines. Additionally, we also report statistical significance $p < 0.05$ and $p < 0.01$ when compared against the FA-NMT models that have two separate attention layers from the decoder to the encoder. Overall, the SA-NMT (shared) model performs the best gaining more than 0.5 BLEU De$\rightarrow$En on wmt16, up to 0.82 BLEU on En$\rightarrow$De wmt17 and 0.64 BLEU En$\rightarrow$Ru direction over a competitive NMT baseline. The results show that structured attention is useful when translating from English to languages that have long-distance dependencies and complex morphological agreements. We also see that the gain is marginal compared to self-attention models (FA-NMT) and not significant. Within FA-NMT models, sharing attention is helpful. Our results also confirm the advantage of having two separate sets of annotations in the encoder when modeling syntax. The hard structured attention model (SA-NMT-hard) performs comparable to the baseline. While this is a somewhat expected result from the hard attention model, we will show in the next section (§6) that the quality of induced trees from hard attention is far better than the soft ones.

<table>
<thead>
<tr>
<th>Model</th>
<th>Shared</th>
<th>De$\rightarrow$En wmt16</th>
<th>Ru$\rightarrow$En wmt17</th>
<th>En$\rightarrow$De wmt16</th>
<th>En$\rightarrow$Ru wmt17</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMT</td>
<td>-</td>
<td>33.16</td>
<td>28.94</td>
<td>30.17</td>
<td>29.92</td>
</tr>
<tr>
<td>FA-NMT</td>
<td>yes</td>
<td>33.55</td>
<td>29.43</td>
<td>30.22</td>
<td>30.09</td>
</tr>
<tr>
<td></td>
<td>no</td>
<td>33.24</td>
<td>29.00</td>
<td>30.34</td>
<td>29.98</td>
</tr>
<tr>
<td>SA-NMT-1set</td>
<td>-</td>
<td>33.51</td>
<td>29.15</td>
<td>30.34</td>
<td>30.29</td>
</tr>
<tr>
<td>SA-NMT-hard</td>
<td>yes</td>
<td>33.38</td>
<td>28.96</td>
<td>29.98</td>
<td>29.93</td>
</tr>
<tr>
<td></td>
<td>no</td>
<td>33.18</td>
<td>29.19</td>
<td>30.15</td>
<td>30.17</td>
</tr>
</tbody>
</table>

5 Attention and Gate Activation Visualization

Figure 4 shows a sample visualization of structured attention models trained on En$\rightarrow$De data. It is worth noting that the shared SA-NMT model (Figure 4a) and the hard SA-NMT model (Figure 4b) capture similar structures of the source sentence. We hypothesize that when the objective function requires syntax, the induced trees are more consistent unlike those discovered by a semantic objective (Williams et al., 2017). Both models correctly identify that the verb is the head of pronoun (hope$\rightarrow$I, said$\rightarrow$she). While intuitively it is clearly beneficial to know the subject of the verb when translating from English into German, the model attention is still somewhat surprising because long distance dependency phenomena are less common in English, so we would expect that a simple content based addressing (i.e standard attention mechanism) would be sufficient in this translation.

We now turn to the question of when does the target LSTM need to access source side syntax. We investigate this by analyzing the gate activations of our best model, SA-NMT (shared). At time step
I hope that this year something will be seen to happen,
she said.

(a) SA-NMT (shared) attention.

(b) SA-NMT with hard structured attention.

Figure 4: A visualization of attention distributions over head words. y-axis shows the head words. Darker color means higher attention weights.

Figure 5: Visualization of gate norm. Best viewed in color.

$z_t = \| \sigma(W_g h_{t-1}) \|_2$ (23)

The activation norm $z_t$ allows us to see how much syntactic information flows into the decoder. We observe that $z_t$ has its highest value when the decoder is about to generate a verb while it has its lowest value when the end of sentence token is predicted. Figure 5 shows some examples of German target sentences. The darker colors represent higher activation norms and bold words indicate the highest activation norms when those words are being predicted.

It is clear that translating verbs requires knowledge of syntax. We also see that after verbs, the gate activation norms are highest at nouns Zeit (time), Mut (courage), Dach (roof) and then tail off as we move to function words which require less context to disambiguate. Below are the frequencies with which the highest activation norm in a sentence is applied to a given part-of-speech tag on newstest2016. We include the top 10 most common activations. It is important to note that this distribution is dramatically different than a simple frequency baseline.

<table>
<thead>
<tr>
<th>VERB</th>
<th>NOUN</th>
<th>AUX</th>
<th>ADP</th>
<th>PUNCT</th>
<th>ADJ</th>
<th>DET</th>
<th>PART</th>
<th>PROPN</th>
<th>ADV</th>
</tr>
</thead>
<tbody>
<tr>
<td>1022</td>
<td>636</td>
<td>193</td>
<td>189</td>
<td>184</td>
<td>170</td>
<td>167</td>
<td>95</td>
<td>75</td>
<td>71</td>
</tr>
</tbody>
</table>

6 GRAMMAR INDUCTION

NLP has longed assumed hierarchical structured representations were important to understanding language. In this work, we have borrowed that intuition to inform the construction of our model (as previously discussed). We feel it is important to take a step beyond a comparison of aggregate model performance and investigate whether the internal latent representations discovered by our models share properties previously identified within linguistics and if not, what important differences exist.

We investigate the interpretability of our model’s representations by: 1) A quantitative attachment accuracy and 2) A qualitative comparison of the underlying grammars.
Table 3: Directed and Undirected (DA/UA) accuracies of our models on both English and German data as compared to branching baselines. Punctuations are removed during the evaluation.

<table>
<thead>
<tr>
<th></th>
<th>FA</th>
<th></th>
<th>SA</th>
<th></th>
<th>Baseline</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>no-shared</td>
<td>shared</td>
<td>no-shared</td>
<td>shared</td>
<td>hard</td>
<td>L</td>
</tr>
<tr>
<td>EN (→de)</td>
<td>17.0/25.2</td>
<td>27.6/41.3</td>
<td>23.6/33.7</td>
<td>27.8/42.6</td>
<td>31.7/45.6</td>
<td>34.0/40.5</td>
</tr>
<tr>
<td>EN (→ru)</td>
<td>35.2/48.5</td>
<td>36.5/48.8</td>
<td>12.8/25.5</td>
<td>33.1/48.9</td>
<td>33.7/46.0</td>
<td>34.4/45.6</td>
</tr>
<tr>
<td>DE (→en)</td>
<td>21.1/33.3</td>
<td>20.1/33.6</td>
<td>12.8/22.5</td>
<td>21.5/38.0</td>
<td>26.3/40.7</td>
<td>34.4/42.8</td>
</tr>
<tr>
<td>RU (→en)</td>
<td>23.2/38.1</td>
<td>26.3/43.0</td>
<td>21.8/37.5</td>
<td>26.5/44.3</td>
<td>22.5/36.6</td>
<td>32.9/47.3</td>
</tr>
</tbody>
</table>

Our results both corroborate and refute previous work (Hashimoto & Tsuruoka, 2017; Williams et al., 2017). We agree and provide stronger evidence that syntactic information can be discovered via latent structured attention, but we also present preliminary results that indicate that conventional definitions of syntax may be at odds with task specific performance.

6.1 EXTRACTING A TREE

For extracting non-projective dependency trees, we use Chu-Liu-Edmonds algorithm (Chu & Liu, 1965; Edmonds, 1967). First, we must collapse BPE segments into words. Assume the $k$-th word corresponds to BPE tokens from index $u$ to $v$. We obtain a new matrix $\hat{\phi}$ by summing over $\phi_{i,j}$ that are the corresponding BPE segments.

$$\hat{\phi}_{i,j} = \begin{cases} 
\phi_{i,j} & \text{if } i \notin [u, v] \land j \notin [u, v] \\
\sum_{l=u}^{v} \phi_{l,t} & \text{if } j = k \land i \notin [u, v] \\
\sum_{l=u}^{v} \phi_{l,j} & \text{if } i = k \land j \notin [u, v] \\
\sum_{l=u}^{v} \sum_{h=u}^{v} \phi_{l,h} & \text{otherwise}
\end{cases}$$

(24)

6.2 GRAMMATICAL ANALYSIS

We compute unlabeled directed and undirected attachment accuracies of our predicted trees on “gold” annotations from Universal Dependencies (UD version 2) dataset\(^3\). Our four model settings in addition to left and right branching baselines are presented in Table 3. The results indicate the basic structured attention models capture headedness at least as well as baselines, but that hard attention actually boosts performance by 4-5 percent across the board. This reflects our hypothesis that capturing interesting structures beyond semantic co-occurrence requires the models to take discrete actions. This corroborates previous work (Choi et al., 2017; Yogatama et al., 2017) which has shown that non-trivial structures are learned by using REINFORCE (Williams, 1992) or Gumbel-softmax trick (Jang et al., 2016) to backprop through discrete units. Our approach also outperforms that of Hashimoto & Tsuruoka (2017) despite our model lacking access to additional resources like part-of-speech tags.

Dependency Accuracies While SA-NMT-hard model gives the best directed attachment scores on both German and English, the BLEU scores of this model are below other SA-NMT models as shown in table 2. The lack of correlation between syntactic performance and NMT contradicts the intuition of previous work and actually suggests that useful structures learned in service of a task might not necessarily benefit from or correspond to known linguistic formalisms.

\(^3\)http://universaldependencies.org
Table 4: Most common grammar rules and their production percentages in EN and DE. English’s strict left branching structure makes it difficult to outperform, but we see substantial gains by our approach on the more syntactic elements of language (e.g. DET/ADJ/ADP attachments). For EN, we use en→ru systems.

<table>
<thead>
<tr>
<th>Rule</th>
<th>EN</th>
<th>DE</th>
</tr>
</thead>
<tbody>
<tr>
<td>gold</td>
<td>left</td>
<td>SA</td>
</tr>
<tr>
<td>VERB → NOUN</td>
<td>25.1</td>
<td>9.5</td>
</tr>
<tr>
<td>→ PRON</td>
<td>18.6</td>
<td>24.9</td>
</tr>
<tr>
<td>→ ADV</td>
<td>9.1</td>
<td>9.9</td>
</tr>
<tr>
<td>→ VERB</td>
<td>12.8</td>
<td>5.4</td>
</tr>
<tr>
<td>NOUN → DET</td>
<td>23.2</td>
<td>19.7</td>
</tr>
<tr>
<td>→ ADP</td>
<td>17.2</td>
<td>14.3</td>
</tr>
<tr>
<td>→ NOUN</td>
<td>18.4</td>
<td>20.4</td>
</tr>
<tr>
<td>→ ADJ</td>
<td>13.9</td>
<td>13.7</td>
</tr>
</tbody>
</table>

Qualitative Grammar Analysis  We should obviously note that the model’s strength shows up in the directed but not the undirected attention. This begs the question as to whether there are basic structural elements the grammar has decided not to attend to or if all constructions are just generally weak. We qualitatively analyzed the learned grammars as a function of dependency productions between universal part-of-speech tags in table 4.

7 Conclusion

We have proposed a structured attention encoder for NMT. Our models show significant gains in performance over a strong baseline on standard WMT benchmarks. The models presented here do not access any external information such as parse-trees or part-of-speech tags. We show that our models induce dependency trees over the source sentences that systematically outperform baseline branching and previous work. We find that the quality of induced trees (compared against gold standard annotations) is not correlated with the translation quality.

References


