Scheduled Multi-Task Learning: From Syntax to Translation

Eliyahu Kiperwasser*
Computer Science Department
Bar-Ilan University
Ramat-Gan, Israel
elikip@gmail.com

Miguel Ballesteros
IBM Research
1101 Kitchawan Road, Route 134
Yorktown Heights, NY 10598. U.S
miguel.ballesteros@ibm.com

Abstract

Neural encoder-decoder models of machine translation have achieved impressive results, while learning linguistic knowledge of both the source and target languages in an implicit end-to-end manner. We propose a framework in which our model begins learning syntax and translation interleaved, gradually putting more focus on translation. Using this approach, we achieve considerable improvements in terms of BLEU score on relatively large parallel corpus (WMT14 English to German) and a low-resource (WIT German to English) setup.

1 Introduction

Neural Machine Translation (NMT) (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014; Bahdanau et al., 2014) has recently become the state-of-the-art approach to machine translation (Bojar et al., 2016). One of the main advantages of neural approaches is the impressive ability of RNNs to act as feature extractors over the entire input (Kiperwasser and Goldberg, 2016), rather than focusing on local information. Neural architectures are able to extract linguistic properties from the input sentence in the form of morphology (Belinkov et al., 2017) or syntax (Linzen et al., 2016).

Nonetheless, as shown in Dyer et al. (2016) and Dyer (2017), systems that ignore explicit linguistic structures are incorrectly biased and they tend to make overly strong linguistic generalizations. Providing explicit linguistic information (Dyer et al.,

2016; Kuncoro et al., 2017; Niehues and Cho, 2017; Sennrich and Haddow, 2016; Eriguchi et al., 2017; Aharoni and Goldberg, 2017; Nadejde et al., 2017; Bastings et al., 2017; Matthews et al., 2018) has proven to be beneficial, achieving higher results in language modeling and machine translation.

Multi-task learning (MTL) consists of being able to solve synergistic tasks with a single model by jointly training multiple tasks that look alike. The final dense representations of the neural architectures encode the different objectives, and they leverage the information from each task to help the others. For example, tasks like multiword expression detection and part-of-speech tagging have been found very useful for others like combinatory categorical grammar (CCG) parsing, chunking and super-sense tagging (Bingel and Søgaard, 2017).

In order to perform accurate translations, we proceed by analogy to humans. It is desirable to acquire a deep understanding of the languages; and, once this is acquired it is possible to learn how to translate gradually and with experience (including revisiting and re-learning some aspects of the languages). We propose a similar strategy by introducing the concept of Scheduled Multi-Task Learning (Section 4) in which we propose to interleave the different tasks.

In this paper, we propose to learn the structure of language (through syntactic parsing and part-of-speech tagging) with a multi-task learning strategy with the intentions of improving the performance of tasks like machine translation that use that structure and make generalizations. We achieve considerable improvements in terms of BLEU score on a relatively large parallel corpus (WMT14 English to Ger-

---

* Work carried out during summer internship at IBM Research.

man) and a low-resource (WIT German to English) setup. Our different scheduling strategies show interesting differences in performance both in the low-resource and standard setups.

2 Sequence to Sequence with Attention

Neural Machine Translation (*NMT*) (Sutskever et al., 2014; Bahdanau et al., 2014) directly models the conditional probability $p(y|x)$ of the target sequence of words $y = <y_1, \ldots, y_T>$ given a source sequence $x = <x_1, \ldots, x_S>$. In this paper, we base our neural architecture on the same sequence to sequence with attention model; in the following we explain the details and describe the nuances of our architecture.

2.1 Encoder

We use bidirectional LSTMs to encode the source sentences (Graves, 2012). Given a source sentence $x = <x_1, \ldots, x_m>$, we embed the words into vectors through an embedding matrix W^S, the vector of the i-th word is $W^S x_i$. We get the representations of the i-th word by summarizing the information of neighboring words using bidirectional LSTMs (Bahdanau et al., 2014),

$$h_i^F = LSTM^F(h_{i-1}^F, \; W^S x_i) \quad (1)$$
$$h_i^B = LSTM^B(h_{i+1}^B, \; W^S x_i). \quad (2)$$

The forward and backward representation are concatenated to get the bi-directional encoder representation of word i as $h_i = [h_i^F, h_i^B]$.

2.2 Decoder

The decoder generates one target word per time-step, hence, we can decompose the conditional probability as

$$\log \, p(y|x) = \sum_j p(y_j|y_{<j}, x). \quad (3)$$

The decoding procedure consists of two main processes: attention and LSTM based decoding. The attention mechanism calculates the weights (α_i) for each source word based on the words translated/decoded so far. The model gives higher weight to words that are more relevant to decode the next word in the sequence. This is based on the words

decoded so far represented by the decoder state (d_j), and the encoder representation of the sentence (h_i). Concretely, we use dot attention (Luong et al., 2015b) to calculate the attention weights. More formally, α_i is calculated as follows:

$$e_i = d_j^\top h_i \quad (4)$$
$$\alpha_i = \frac{\exp(e_i)}{\sum_k \exp(e_k)}. \quad (5)$$

A vector representation (c_j) capturing the information relevant to this time-step is computed by a weighted sum of the encoded source vector representations using α values as weights.

$$c_j = \sum_i \alpha_i \cdot h_i. \quad (6)$$

Given the sentence representation produced by the attention mechanism (c_j) and the decoder state capturing the translated words so far (d_j), the model decodes the next word in the output sequence. The decoding is done using a multi-layer perceptron which receives c_j and d_j and outputs a score for each word in the target vocabulary:

$$g_j = \tanh(W_{Dec}^1 d_j + W_{Att}^1 c_j) \quad (7)$$
$$u_j = \tanh(g_j + W_{Dec}^2 d_j + W_{Att}^2 c_j) \quad (8)$$
$$p(y_j|y_{<j}, x) \approx softmax(W_{out} u_j + b_{out}). \quad (9)$$

3 Many Tasks One Sequence to Sequence

Sequence to sequence models have been used for many tasks such as: machine translation (Sutskever et al., 2014; Bahdanau et al., 2014), summarization (Rush et al., 2015) and syntax (Vinyals et al., 2015). Several recent works have shown that parameter sharing between multiple sequence to sequence models that aim to solve different tasks may improve the accuracy of the individual tasks (Kaiser et al., 2017; Luong et al., 2015a; Zoph and Knight, 2016; Niehues and Cho, 2017; Bingel and Søgaard, 2017, inter-alia).

We apply a simple yet effective approach to learn multiple tasks using a single sequence to sequence model inspired by Ammar et al. (2016). All tasks share a common output vocabulary and generate terms according to (3). We learn multiple tasks simultaneously by prepending a special task embedding vector to the target. The task vector symbolizes the task we are focusing on. The model can
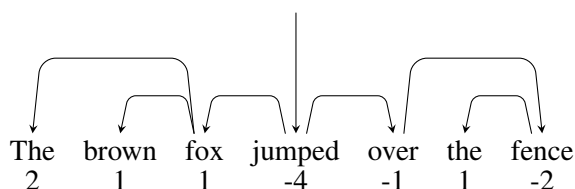
226

Figure 1: Illustration of the encoding of an unlabeled parsing tree into a sequence of distances. The first row contains the sentence (source) and its parse tree, and the second row contains the matching distances sequence (target).

solve each of the tasks it was trained on by priming the decoder with the token of each task. Johnson et al. (2017) suggested to prepend a special embedding vector according to the desired target language. In contrast to our approach, they prepend the vector to the encoder and not to the decoder.

We apply this methodology to jointly learn the multiple tasks, however many of the tasks are not of sequential nature (such as dependency parsing in which the output should be a well-formed dependency tree (Hudson, 1984; Melčuk, 1988)). We fit those into our sequence to sequence model in order to enrich the representation of other tasks, and increase the potential information flow between the tasks. In what follows, we show which tasks (and how we linearize them) we solve jointly using our model and how we apply sequence to sequence modeling to those tasks.

Part-Of-Speech Tagging   Given a sentence and its part-of-speech annotation, we convert the task to translating between the sentence (as the source sequence) and the given part-of-speech tags as the target. A similar approach was suggested by Niehues and Cho (2017).

Unlabeled Dependency Parsing   An unlabeled dependency tree annotation can be viewed as a sequence of heads, where for every node there is a unique incoming edge, that is, a single matching head. We convert the tree by scanning the sentence from left to right, and outputting the distance of each word to its head. We then convert the task to translating between the original sentence and the resulting sequence describing the unlabeled dependency tree
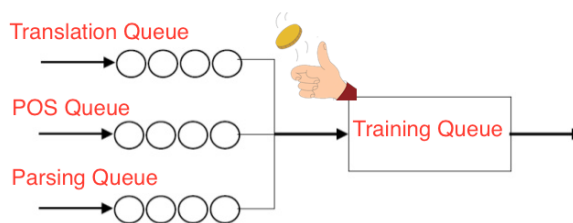

Figure 2: Illustration of Scheduled Multi-Task Learning approach. This figure contains a flow chart describing how the training examples from multiple sources are gathered into a single training queue. The chart starts from multiple queues, each containing training examples belonging to different tasks (possibly from different datasets). Using a coin toss we choose the next queue to take the following training example from. The probability of each of the queues to be selected is determined by the scheduler.

(See Figure 1). Sequence of distances is an invertible representation of the sequence of heads, which is equivalent to an unlabeled tree. In contrast to a sequence of heads, learning a sequence of distances is able to generalize to sentences of arbitrary length (including length which are not seen or rarely seen in the training corpus). Distance to the syntactic heads has also been shown to be an effective feature when parsing sentences (McDonald et al., 2005).

Predicting Dependency Relations-Labeled Dependency Parsing   Similarly to the conversion of the unlabeled dependency tree to a sequence, we scan all the words in the sentence from the beginning to the end. For each word encountered, we output the label of the dependency arc connecting it with its matching head word. We, therefore, learn to translate between the original sentence and the resulting sequence of dependency labels.

Machine Translation   Similarly to Sutskever et al. (2014) and Bahdanau et al. (2014), we use sequence to sequence to translate between a sentence written in a source language and a sentence written in a target language.

4   Scheduled Multi-Task Learning

In order to produce accurate translations, neural machine translation systems have to learn syntax in

order to generate grammatically correct sentences. Furthermore, translation systems have to disambiguate different parts-of-speech on the source side sentence, since a different part-of-speech can result in different translations. There are many sets of parameters able to capture the training data when employing LSTM (RNN) models. This applies to sequence to sequence models with attention. Each set of parameters provides a different level of generalization (Reimers and Gurevych, 2017). As suggested by Dyer (2017), representations learned by the network do not capture the linguistic properties, and they are biased to make overly strong linguistic generalizations.

Providing "guidance" to the sequence to sequence network at the beginning focusing it on a representation enriched with linguistic knowledge, such as syntax or part-of-speech tagging, helps it obtain information necessary for converging to a more general solution. We suggest interleaving the learning of the syntax and translation tasks, and gradually decrease the weight of the syntactically oriented tasks (auxiliary tasks). This enables the model to forget about the syntax examples and to put more focus on fitting the translation task as the training progresses.

Our approach, Scheduled Multi-Task Learning (SMTL), is a semi-supervised learning approach that generalizes the above scheme. Scheduled Multi-Task Learning continuously interleaves between three well-known previous methods: Multi-task learning, Pre-training, and Fine-tuning.

Multi-Task Learning (*MTL*) (Caruana, 1997) solves synergistic tasks while maximizing the number of shared parameters. Sharing parameters for multiple tasks may increase the accuracy in tests for the individual tasks, thanks to representation bias which captures a more regularized representation fitted to multiple tasks (Baxter, 2000) and using information from one task as hints to the other tasks (Abu-Mostafa, 1990). In case of independence between the features of the multiple tasks learned, we assume that enforcing the representation to accommodate multiple tasks can result in a drop in accuracy compared to the accuracy of each task learned separately (Caruana, 1997; Bingel and Søgaard, 2017).

Pre-Training (Collobert et al., 2011) is a widely used approach (Goldberg, 2017) which initializes the parameters with the parameters used to solve a somewhat related task. Similarly, Fine-Tuning uses a small annotated in-domain corpus and a large annotated out-of-domain corpus to estimate parameters. We first learn using the large out-of-domain corpus and once that is finished, we continue learning (fine-tuning) on the in-domain corpus. This is a common approach for transfer learning (Yosinski et al., 2014). A related approach is to start with a pre-trained neural network model and fine-tune only the final layers in order to keep the coarse features detected for the previous task (Hinton and Salakhutdinov, 2006; Erhan et al., 2010). Both approaches, facilitate encoding useful information from related tasks (Pre-training) or data-sets (Fine-tuning) without demanding that the representation accommodate both tasks, and can be viewed as regularization (Caruana, 1997).

Our Scheduled Multi-Task Learning approach unifies the above methods into a single framework. This framework contains multiple queues, where each queue contains the training examples belonging to a specific pair of tasks and datasets. In order to pick the next training example, we stochastically pick a queue (q) with time-dependent probability (p_q^t) and then we get the next example from the chosen queue (Figure 2).

The probabilities (p_q^t) change as the training progresses according to a *Schedule*. The Schedule could, for example, give a high probability at the beginning of the training process to some task (e.g. part-of-speech tagging) and gradually decrease the probability in favor of another task (e.g. translation). The latter schedule resembles the pre-training approach at the beginning by, later in the process, progressing to multi-task learning approach. Such a schedule enables harnessing hints from related tasks and also enforces a soft representation bias at the beginning of the training. This contrasts with previous schemes, which either they used solely pre-training and therefore were not able to benefit from the representation bias, or they used solely multi-task learning and were not able to tweak the representation bias.

We aim to improve generalization over a specific task and dataset (queue) using examples from related tasks and datasets. We suggest three schedulers to do so: *Constant Scheduler*, *Exponential*

(a) Constant Schedule   (b) Exponential Schedule   (c) Sigmoid Schedule
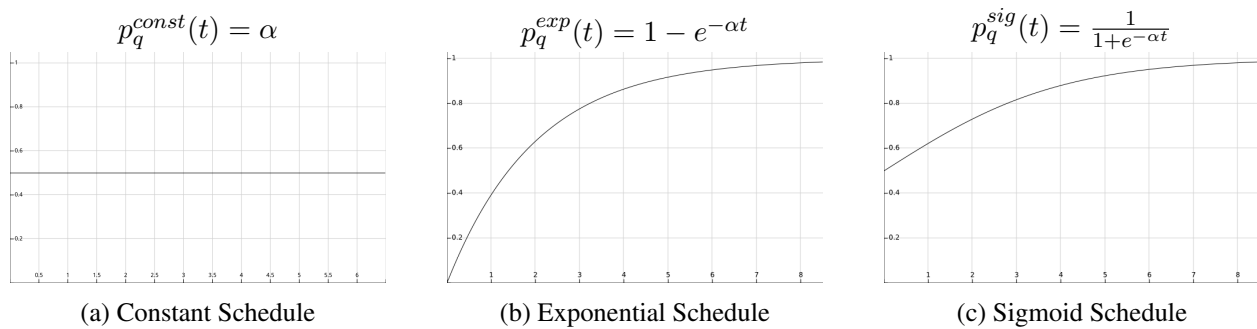
Figure 3: Illustration of different scheduling strategies determining the probability of the next training example to be picked from each of the multiple tasks we learn. Each sub-plot in the figure matches a different scheduling strategy (with α set to 0.5). The sub-plot describes the probability (p, y-axis) of the task we wish to improve (q) using Scheduled Multi-Task Learning as a function of the number of epochs trained (t, x-axis) so far. The remaining probability is uniformly distributed among the rest of the tasks.

Scheduler, and *Sigmoid Scheduler* (Figure 3). As input, the schedulers receive the fraction of training epochs done so far ($t = {}^{Sent}/_{\|Corpus\|}$), and a hyper-parameter (α) determining the slope of the scheduler. Given slope parameter (α) and the epoch number, the chosen scheduler depicts a multinomial distribution for choosing each of the queues as the source of the next training example.

Constant Scheduler We assign constant probability to the queue we focus on and divide the rest of the probability uniformly between remaining queues ($p_q(t) = \alpha$). This is similar to previous Multi-Task Learning approaches (Caruana, 1997).

Exponential Scheduler We assign exponentially increasing probability to the queue we focus on and divide the rest of the probability uniformly between remaining queues ($p_q(t) = 1 - e^{-\alpha t}$). This approach starts by only looking at the training from all the tasks besides the task that we wish to focus on, and it later tunes the parameters based solely on the main task (resembling pre-training and fine-tuning).

Sigmoid Scheduler We assign probability to the queue we focus on using a sigmoid and divide the rest of the probability uniformly between remaining queues ($p_q(t) = \frac{1}{1+e^{-\alpha t}}$). This approach starts by looking at all tasks (resembling MTL), and it later tunes the parameters based solely on the main task we wish to focus on.

5 Experimental Setup

We evaluate the effectiveness of our models for a low-resource setting and a standard setting. Translation performances are reported in case-sensitive BLEU (Papineni et al., 2002). We report translation quality using tokenized[1] BLEU comparable with existing Neural Machine Translation papers.

Our experiments are centered around the translation task. We aim to determine whether other syntactically oriented tasks can improve translation and vice versa. Each task is presented in a sequence to sequence manner (as described in Section 3). A single sequence to sequence with attention model is used to solve all tasks (all the parameters are shared between the different tasks).

5.1 Data

We train the byte-pair encoding model (Sennrich et al., 2016) for the translation parallel corpus and apply it to all the data (including non-translation data).

Syntax For English, we extract part-of-speech tagging, dependency heads and labels from the Penn tree-bank (Marcus et al., 1993) with Stanford Dependencies[2]. For German, we extract them from TIGER tree-bank (Brants et al., 2002).[3] Both tree-banks are annotated by experts and contain the

---

[1] All texts are tokenized with `tokenizer.perl` and BLEU scores are computed with `multi-bleu.perl`.

[2] Training: 02-21. Development: 22. Test: 23.

[3] German CoNLL 2009 dataset (Hajič et al., 2009).

229

gold annotations for dependency parsing and part-of-speech tags.

Given the language, we extract three datasets from the relevant tree-bank. We extract parallel corpus of sentences and their gold part-of-speech annotations. The same is done in order to extract a dataset of the unlabeled distances and the dependency labels.

Translation In order to simulate low-resource translation tasks, we used 4M tokens of the WIT corpus (Cettolo et al., 2012) for German to English as training data. We used *tst2012* for validation and *tst2013* for testing, provided by the International Workshop on Spoken Language Translation (IWSLT). Byte-pair encoding is applied, resulting in a vocabulary of 29937 tokens in the source side and 21938 tokens in the target side.

For standard translation setting, we use WMT parallel corpus (Buck et al., 2014) with 4.5M sentence pairs (we translate from English to German). We use *newstest2013* (3000 sentences) as the development set to select our hyper-parameters, and *newstest2014* for testing. Note that we use the same (MT) development sets to select the hyper-parameters of the syntactically oriented tasks. After byte-pair encoding is applied, it results in a vocabulary of 59937 tokens in the source side and 63680 tokens in the target side.

We only used training examples shorter than 60 words per sentence. We also filter out pairs where the target length is more than 1.5x times the source length.

5.2 Training Details

We use mini-batching that limits the number of words in the mini-batch instead of the number of sentences (Morishita et al., 2017). We limit the mini-batch size to 5000 words. Based on the scheduler we sample, the dataset to draw training examples from, and add it to the mini-batch until the word limit is reached. *In contrast* to other approaches (Luong et al., 2015a; Zoph and Knight, 2016), our mini-batch is not separated by tasks and often contains examples from multiple tasks. We shuffle each dataset at the beginning of the training, and after the model has been trained on all the source and target pairs belonging to the dataset(s).

We use a two layer stacking BILSTM for the de-

coder, and a single layer BILSTM for the encoder. For the low-resource setting, the number of dimensions of the LSTM and the word embedding is set to 250. For the standard setting, the number of dimensions is set to 500. The dimensionality in the standard setting is set to 500 (instead of 1000), in order to enable quick convergence and thereby examine our approach in many different combinations. The weight updates were determined using the unbiased Adam algorithm (Kingma and Ba, 2014).

We used 0.5 as the scheduler's slope (α) (see Section 4) for all our experiments. We use beam search decoding (of size 5) when decoding the test results. For all tasks (including dependency parsing and part-of-speech tagging), we choose the model that maximizes the BLEU score between the reference development corpus and the system prediction on that corpus. For each scheduler and combination of tasks, we report the test score of the model achieving the best development score of three single runs (each with different random initialization).

Our code is implemented in C++, using the DyNet framework (Neubig et al., 2017). When running on a single GPU device Tesla K80, it takes 5-7 days to completely train a model with 4.5 million sentence pairs, and 12 hours for the low resource setup (4M tokens).

6 Results

We show the base performance of each task using our Many Tasks One Sequence to Sequence model (subsection 6.1). We explore multiple combinations of those concurrently learned using Scheduled Multi-Task Learning (subsection 6.2). We explore (subsection 6.3) different slope parameter (α) values (see Section 4) with the intention of optimizing machine translation by leveraging the additional tasks. Finally, we compare our architecture with an architecture that uses separate decoders for each task (Section 6.4) with a focus on machine translation.

6.1 Auxiliary Tasks

We use dependency parsing and part-of-speech tagging as auxiliary tasks. Our method utilizes BiLSTM features for syntax as proposed by Kiperwasser and Goldberg (2016) and attention proposed by Dozat and Manning (2016), however ours does

not impose any tree structure constraints since it is the architecture for translation described in Section 2. The model does not even contain the length of the sentence as a hard constraint, meaning that it can arbitrarily output a shorter/longer sequence.[4] Although no structural constraints were imposed, our sequence to sequence model is able to obtain a decent parsing result.[5] The model achieves 86.99 UAS for English Penn tree-bank with Stanford Dependencies,[6] and 80.28 UAS for the German TIGER tree-bank when the model is only trained to predict the sequence of distances to head as described in Section 3. This is below the best results achieved by state-of-the-art parsers, that are already around 95 for English (Dozat and Manning, 2016; Kuncoro et al., 2017), and around 90 for the same German dataset (Andor et al., 2016; Kuncoro et al., 2016; Bohnet and Nivre, 2012). As a side product of our research, we show that dependency parsing can be approached via a sequence to sequence with an attention mode commonly used for neural machine translation with linearized (using sequences of head distances) dependency trees. Note that, in this case, the models are solely trained on predicting the sequence of distances to the head and are not trained to predict the sequence of dependency labels.

For part-of-speech tagging, we use the same sequence to sequence with attention architecture presented in Section 2. Our model uses BiLSTM encodings, in a similar way as proposed by Wang et al. (2015) for part-of-speech tagging. Similarly as in parsing (see above), we do not force one part-of-speech per word and do not force the model to scan the sentence linearly nor do we add any hard constraints on the length. Even without these constraints, the model achieves accuracy of 95.07 for English Penn tree-bank and 95.41 for German

TIGER treebank, which is lower than the best systems that achieve results above 97 (Andor et al., 2016; Bohnet and Nivre, 2012) for both languages. We use the same datasets as in the parsing task.

Note that both for part-of-speech tagging and dependency parsing, our models are trained with byte-pair encoding (BPE) in the input side (Sennrich et al., 2015), meaning that there are usually more tokens in the input than in the output (which has exactly one label or a token representing the distance to head per word). For the single-task models we also use 250 dimensions for our network (word embeddings, hidden dimensions and LSTM input dimensions) for German and 500 dimensions for English.

6.2 Translation Task

We start from our baseline system which achieves results which are comparable (see Tables 1 and 4) to the ones reported by Bahdanau et al. (2014) on the standard setting (WMT), and Niehues and Cho (2017) on the low-resource setting (IWSLT). We examine the effect of Scheduled Multi-Task Learning on the translation quality compared to the baseline system with a constant value of the slope parameter (α) set to 0.5.[7] We also show that amount of representation bias the models chose to obtain by testing each model on each of the auxiliary tasks.

As in part-of-speech tagging and dependency parsing (both predicting a sequence of heads and dependency labels, as separate tasks. This is the reason why we report LAS), we use BPE encoding both in target and source. We use 250 dimensions for the low-resource setting (IWSLT) and 500 dimensions for the standard setting (WMT).

Low-Resource Setting In a low-resource setting, we witness a significant increase in translation quality when doing basic multi-task learning (with the constant scheduler) with syntactic auxiliary tasks (Table 1). We attribute this to the additional linguistic information which is difficult to learn from a low-resource setting. The latter can be observed in Table 1 which shows an increase of roughly 2.7 BLEU when adding part-of-speech information and 1.85 BLEU when adding dependency parsing.

---

[4]All evaluation metrics penalize sequences of the wrong length.

[5]The parsing only model (without MTL) was trained solely on the unlabeled dependency arcs. Full parsing model that was used in conjunction with other tasks was trained as separate tasks (in an MTL manner) on both unlabeled arcs and their labels.

[6]By increasing the dimensionality of the network for the English parsing task, we achieve results around 90 UAS, but in Table 4 we report results with 500 dimensions since it is the one used in the multi-task learning scenario with the WMT data (see Section 6.2).

[7]In Section 6.3 we explore different slope parameter (α) values for the same task.

The baseline (constant) multi-task learning scheduler reaches better translation quality than the sigmoid and exponential scheduler. We hypothesize that in a low-resource setting a strong representation bias incorporating linguistic knowledge helps to build generalized representation which cannot be obtained from a relatively small parallel corpus.

We evaluate the dependency parsing scores and the part-of-speech tagging accuracy of the models tuned to perform translation on the held-out development set. The percentage of correctly predicted unlabeled arcs by MTL is no more than 10 UAS points worse compared to the models that are solely train to parse or to tag, and they are very close for the Constant Scheduler. Note that the models are optimized to perform translation, however they are still able to parse sentences with a reasonable accuracy. MTL models are also better at translation than models trained on the vanilla translation data. This means that the attentional model of translation is benefiting from the syntactic information, and therefore chooses to learn parameters close to the syntactically oriented tasks, even though there are no constraints forcing it to do so.

As mentioned above, the automatic scores show a significant improvement over the NMT system that only sees the parallel sentences. In Table 2, we show some randomly picked examples from the IWSLT development data in order to show how each of the systems performed. We include Google web[8] system to see a comparison with a state-of-the-art system that is probably trained with more data. Note that in the low-resource data we only have 300k sentence pairs. We selected the output of the systems with highest score in each category (NMT Only, NMT+POS with Constant Scheduler, NMT+Parsing and NMT+POS+Parsing with Exponent Scheduler).

Given that the examples in Table 2 suggest that the SMTL models may be doing a better job at avoiding dropping words we complement our BLEU scores with the METEOR evaluation metric (Lavie and Agarwal, 2007) which is more sensitive to recall. We report METEOR (and fragmentation penalty that captures how well the system produces the correct order of the words) for the models with highest BLEU scores in each category (NMT Only,

NMT+POS with Constant Scheduler, NMT+Parsing and NMT+POS+Parsing with Exponent Scheduler). Table 3 shows the results. Models with the higher BLEU scores also produce higher METEOR scores. In addition it is interesting to see that the fragmentation penalty is higher for the NMT Only model; the NMT Only model only produces 19,768 test words (for the entire test set) while the rest produce longer sentences with more than 20,400 test words. All of this suggests that the additional tasks are helping to avoid dropping parts of the sentence which leads to more adequate outputs.

Standard Setting In the standard-resource setting (Table 4), the exponent scheduler (when using part-of-speech tagging as an auxiliary task) achieves significantly better numbers than the other multi-task learning strategies, and achieves a translation quality that surpasses the base neural translation system (by 0.7 BLEU points). When applying the Constant Scheduler (basic multi-task learning) we see a deduction of at least 1 BLEU point compared to the score of the translation without multi-task learning. We assume that additional out-of-domain linguistic knowledge (such as syntax in the Penn tree-bank) might confuse the linguistic properties that the translation model is inferring from the comparably large machine translation data.

The sigmoid scheduler reaches better translation quality than the constant scheduler by roughly 1 BLEU point (and improves over the base neural translation system) and it improves over the Exponent Scheduler for the tasks that include the parsing objective. This suggests that putting more emphasis on syntax regularizes the model towards capturing linguistic properties (as exponential scheduler does), but that focusing on them as the training continues causes a representation bias which puts focus on out-of-domain data, which, as a result, degrades the translation quality.

Similarly to the low-resource setting, we evaluate the dependency parsing scores and the part-of-speech tagging accuracy of the models tuned to perform translation on the held-out development set. The result for the standard setting shows a drop (of 12 UAS point at most) in the parsing accuracy when trained in a multi-task manner. The accuracy of the part-of-speech tagger improves when using con-

---

[8] https://translate.google.com

232

Scheduler	Tasks	BLEU	POS	UAS	LAS
No MTL	NMT	27.70	–	–	–
	POS	–	95.41	–	–
	Parsing (Unlabeled)	–	–	80.28	–
Constant Scheduler	NMT + POS	**30.4**	93.51	–	–
	NMT + Parsing	28.73	–	79.78	74.25
	NMT + POS + Parsing	29.08	94.80	79.38	74.13
Exponent Scheduler	NMT + POS	30.15	89.05	–	–
	NMT + Parsing	29.37	–	67.60	60.71
	NMT + POS + Parsing	29.55	91.48	72.85	66.44
Sigmoid Scheduler	NMT + POS	30.2	90.74	–	–
	NMT + Parsing	28.78	–	69.26	62.43
	NMT + POS + Parsing	28.93	89.11	65.92	58.46

Table 1: Scheduled Multi-Task learning results for IWSLT German to English translation.

System	Example
Source	Jeden Tag nahmen wir einen anderen Weg , sodass niemand erraten konnte , wohin wir gingen .
Google	Every day we took a different route so no one could guess where we were going.
NMT Only	We took another way for us to guess that no one could guess where we left.
NMT+POS	Every day we took another way so no one could guess where we went.
NMT+Parsing	Every day we took another way that no one could guess where we went.
NMT+POS+Parsing	Every day we took another way that no one could guess where we were going.
Source	Wissen Sie, wie viele Entscheidungen Sie an einem typischen Tag machen ?
Google	Do you know how many decisions you make on a typical day?
NMT Only	You know how many decisions you make on a typical day?
NMT+POS	You know how many decisions you make on a typical day?
NMT+Parsing	Do you know how many decisions you make on a typical day?
NMT+POS+Parsing	Do you know how many decisions you make on a typical day?
Source	Im Winter war es gemütlich, aber im Sommer war es unglaublich heiß.
Google	in winter it was cozy, but in the summer it was incredibly hot.
NMT Only	In winter, it was comfortable, but it was incredibly hot.
NMT+POS	In summer, it was comfortable, but in summer it was incredibly hot.
NMT+Parsing	In the winter, it was comfortable, but in the summer it was incredibly hot.
NMT+POS+Parsing	In the winter, it was comfortable, but in summer it was incredibly hot.

Table 2: Examples from our low-resource IWSLT German to English translation.

System	BLEU	Fragmentation	METEOR
NMT Only	27.7	50.36	30.91
NMT+POS	**30.4**	**49.95**	**31.83**
NMT+Parsing	29.37	50.14	31.43
NMT+POS+Parsing	29.55	50.02	31.56

Table 3: METEOR results for our best scoring systems in comparison with BLEU scores. Fragmentation refers to the fragmentation penalty.

stant and sigmoid schedulers. The part-of-speech accuracy plunges significantly when using the exponential scheduler; and in turn, the translation quality raises by 0.7 BLEU over the baseline model. This suggests that softening the representation bias (by allowing the model to gradually fine-tune on translation) is necessary to improve the translation task. When adding dependency parsing and part-of-speech tagging, we do not see a significant drop in

those auxiliary tasks and also the results for translation does not improve. This might suggest that representation bias is too strict in this case and does not allow the representation to learn beyond the auxiliary tasks.

In order to complement our automatic scores, we performed simple human evaluation, in which an independent German native speaker (who is also proficient in English) scored 50 sentences from 0 to 5 (being 0 exceptionally poor, and 5 excellent); the sentences were randomly shuffled so there is no bias towards the position in which they were presented. The NMT only system achieved a score of 2.54, the best system with part-of-speech tagging only (which is the constant scheduler) achieved 2.68, and both systems that incorporate dependency parsing (NMT+Parsing and NMT+POS+Parsing with the sigmoid scheduler) achieve 2.78 in average. An

Scheduler	Tasks	BLEU	POS	UAS	LAS
No MTL	NMT	19.30	–	–	–
	POS	–	95.07	–	–
	Parsing (Unlabeled)	–	–	86.99	–
Constant Scheduler	NMT + POS	18.29	95.73	–	–
	NMT + Parsing	17.87	–	85.74	81.65
	NMT + POS + Parsing	18.09	96.30	86.58	82.83
Exponent Scheduler	NMT + POS	**20.02**	89.89	–	–
	NMT + Parsing	18.85	–	80.40	74.70
	NMT + POS + Parsing	18.04	94.68	82.54	77.57
Sigmoid Scheduler	NMT + POS	19.21	95.20	–	–
	NMT + Parsing	19.08	–	75.42	69.27
	NMT + POS + Parsing	19.26	94.66	80.33	75.15

Table 4: Scheduled Multi-Task learning results for WMT14 English to German translation.

example output of the systems, also compared to Google, is shown in Table 5; we observe how the system that uses all auxiliary tasks manages to get the gender agreement right for the words *journalist* and *Katie*.

6.3 Scheduler Tuning

We study the impact of different slope parameter (α) values on the translation BLEU score using the low-resource IWSLT corpus. For each scheduler, we train the model (pick the model performing best on the development set) four times with multiple α values and different auxiliary tasks, and average the BLEU score of the decoded test set (Figure 4).

We compare the average result of the Constant Scheduler (Figure 4) against the result of the best performing model on the development set (Table 1). The average result when training with auxiliary tasks (i.e. the Constant Scheduler where α is set to zero) is significantly higher than the result of the best model on the development set (0.7 BLEU points), the matching scores are 28.5 and 27.7 BLEU points. The average score when using the Constant Scheduler with α set to half is greater than the score of the best performing model on the development set. The average result of the constant scheduler setting suggests that multi-task learning helps to mitigate overfitting.

The average results of a model with both parsing and part-of-speech tagging peak when the slope parameter (α) is approximately 1 for both the exponential scheduler (29.43 BLEU) and the sigmoid scheduler (29.55 BLEU). For those schedulers, if the α value is high, the probability of training on the aux-

iliary tasks decreases more rapidly. This suggests that the model needs syntactically oriented synergistic tasks to guide the initial steps to improve convergence; after four epochs the probability of training on an auxiliary task is negligible. The constant scheduler peaks when alpha is 0.2 (yielding an average score of 29.03 BLEU), suggesting that enriching the representation with a small amount of syntactical information helps. This confirms our intuition that syntax is helpful.

Looking at the constant scheduler, which performed best for this dataset (Table 1), we see that the best result is achieved by using parsing as the single auxiliary task (without parts-of-speech). This hints that parsing has potential to help machine translation, even more than part-of-speech tagging with constant scheduler (Niehues and Cho, 2017).

6.4 Architecture Comparison

In order to further validate that the contribution of Scheduled Multi-Task Learning is not limited to our chosen sequence to sequence architecture, we study the impact of our method with a single (and shared across tasks) encoder and the architecture of separate decoders which has already proven to be a very effective multi-task learning scheme (Luong et al., 2015a; Niehues and Cho, 2017). In the latter, each of the decoders is responsible for a different task (i.e. syntax, parts-of-speech, translation, etc.) using a single representation generated by the shared encoder.

In Figure 5, we show the comparison between our Many Tasks One Sequence to Sequence architecture (Section 3) and the architecture of separate decoders

System	Example
Source	In an interview with US journalist Katie Couric , which is to be broadcast on Friday (local time) , Bloom said , ” sometimes life does n't go exactly as we plan or hope for ” .
Google	In einem Interview mit der US-Journalistin Katie Couric, das am Freitag (Ortszeit) ausgestrahlt wird, sagte Bloom: "Manchmal läuft das Leben nicht genau so, wie wir es planen oder erhoffen".
NMT Only	In einem Interview mit der US - Journalist Katie Couric, das am Freitag (Ortszeit) verbreitet werden soll, sagte Bloom, "manchmal geht das Leben nicht genau wie wir planen oder Hoffnung für".
NMT+POS	In einem Interview mit den US - Journalisten Katie Couric, die am Freitag (Ortszeit) ausgestrahlt werden soll, sagte Bloom: "Manchmal ist das Leben nicht genau so, wie wir es planen oder hoffen."
NMT+Parsing	In einem Interview mit dem US - Journalist Katie Couric, der am Freitag gesendet wird (Ortszeit) , sagte Bloom, "manchmal wird das Leben nicht genau so aussehen, wie wir uns vorstellen oder hoffen".
NMT+POS+Parsing	In einem Interview mit US - Journalistin Katie Couric , das am Freitag ausgestrahlt wird (Ortszeit), sagte Bloom: "Manchmal ist das Leben nicht genau so, wie wir planen oder hoffen".

Table 5: Example from our standard English to German translation (WMT).
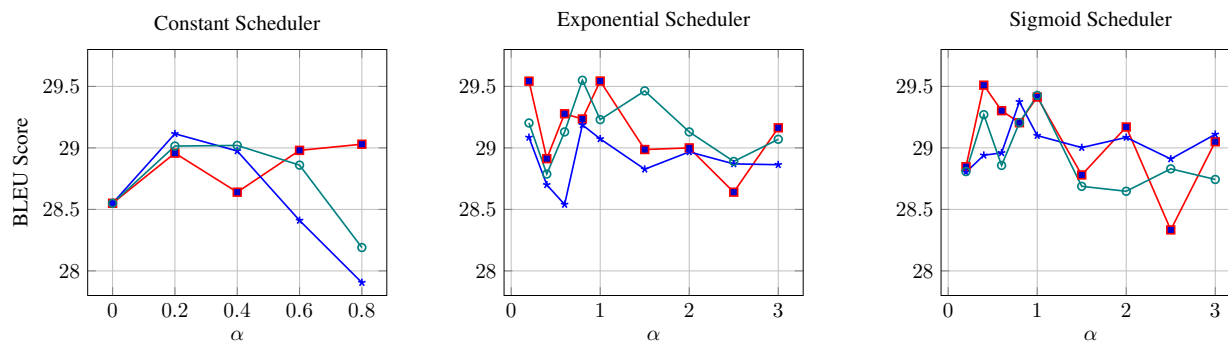

Figure 4: A plot of the BLEU score for the three schedulers over different alpha values. The BLEU score is the average test score of four independent experiments trained on IWSLT training set optimized for maximal score on the IWSLT development set. We use $0 \leq \alpha < 1$ for the constant scheduler, and $0 \leq \alpha \leq 3$ for both the exponential and the sigmoid schedulers. The red line (squares) represents result with part-of-speech tagging as auxiliary task; blue line (asterisks) represents result with parsing as auxiliary task; teal line (circles) represents the results with both tasks as auxiliary tasks.

by using the IWSLT data set. We report BLEU scores as the average test score of four independent experiments for each scheduler and each value of the slope parameter α. The plot shows the average for all schedulers. The best average score for most of the alpha values is greater than the average score without Scheduled Multi-Task Learning (28.5 BLEU). We conclude that scheduled multi-task with syntactic auxiliary tasks is helpful not solely for our architecture, but potentially for other systems as well.

The architecture of separate decoders and a shared encoder peaks at 29.68 BLEU which is higher than the peak score of the shared decoder architecture (29.55 BLEU) by 0.15 BLEU points. The best result of the separate decoders significantly varies as alpha is changed ($\sigma = 0.38$). The result of the shared decoder architecture also varies for different alpha, but in a more subtle manner ($\sigma = 0.21$). This suggests

that the separate decoders architecture is more sensitive to the scheduler used than the shared decoders architecture.

7 Discussion

Scheduled Multi-Task Learning is complementary to other transfer learning methods like pre-training and fine-tuning. It is common to use pre-training in the form of word embeddings (Mikolov et al., 2013; Goldberg, 2017). One advantage of pre-trained word embeddings is the representation of out-of-vocabulary (OOV) words. Through pre-training, OOV words are commonly trained using an early-stopping methodology so their representation remains close to words in the training corpus, thus enabling the model to generalize for unseen words and achieve higher performance in the final task. This constraint limits the flexibility of the optimizer to
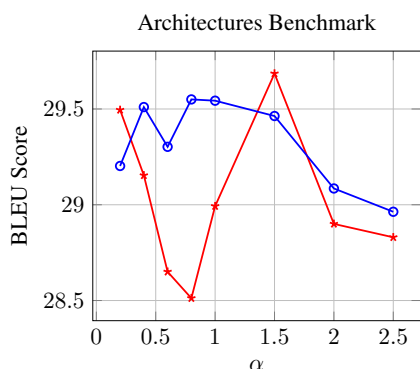
235

Figure 5: A plot of the best BLEU score for each alpha value using our approach (blue line with circle) and separate decoders (red line with asterisks). The BLEU score is the average test score of four independent experiments for each scheduler and each value of alpha trained on IWSLT training set optimized for maximal score on the development set.

choose better word representation for words within the training corpus. Scheduled multi-task learning (and the exponential scheduler in particular) mitigates this problem by allowing the representation of the final task and the auxiliary tasks to be tuned to best fit each other.

The exponential scheduler starts by pre-training the model on an auxiliary task (in our case, part-of-speech tagging and dependency parsing) and gradually puts more focus on our main task (NMT). This enables the model to start with a representation which is able to solve structured prediction tasks containing linguistic knowledge; as the training progresses and the focus is shifted by the scheduler towards the main task, the OOV words representations continue to represent the syntax objective since the auxiliary tasks are less visited but still in use during training. Having embeddings that share the same space enables the model to share information between the tasks, and functions as regularization (Caruana, 1997). The effectiveness of this scheduler is supported by the results (Table 1) showing superior results (on average) on the WIT German to English translation task.

Many approaches have been employing multi-task learning in order to inject linguistic knowledge with great success (Luong et al., 2015b; Niehues and Cho, 2017; Martínez Alonso and Plank, 2017,

inter-alia). The final representation is then adapted to solve multiple tasks, however continuing to fine-tune on solely the main task might result in better accuracy. The latter resembles the Sigmoid Scheduler which starts with multi-task learning and gradually shifts to fine-tuning. The results (Table 4) support that this approach can further benefit multi-task learning systems since it shows superior results (on average) in the WMT14 English to German translation task, although it is still not more superior than the baseline that does not use MTL.

8 Conclusions and Future Work

This paper presents an architecture to perform multi-task learning focusing on the attentional model of translation jointly with linearized dependency parsing and part-of-speech tagging. We show how diverse scheduling strategies perform differently and help to improve the scores in a low-resource setting and a standard setting (bigger dataset). The exponent scheduler achieves the best results on average and the trained models still remember how to perform the auxiliary tasks (part-of-speech tagging and dependency parsing). This means that a key aspect of our models is that they are able to improve the translation accuracy by incorporating syntactically based objectives into the model. Our models report modest dependency parsing and part-of-speech tagging numbers but they clearly learn to perform the tasks; it is worth noting that there is a lack of constraints related to sequence length and correspondence between input tokens and tags/distances which is needed to achieve good parsing scores (Zhang et al., 2017).

We also want to explore another family of schedulers which treats the layers of the neural network differently. For instance, the scheduler can gradually freeze the top LSTM layer of the decoder (by lowering the learning rate), allowing fine-tuning only of the bottom LSTM layer when training for auxiliary tasks. Søgaard and Goldberg (2016) demonstrated the potential of such an approach. Our experiments show that scheduled multi-task learning is very sensitive to the type of scheduler chosen, and many types of schedulers can be explored. We plan to carry out these experiments in the future.

Acknowledgments

Many thanks to Todd Ward, Wael Hamza, Yaser Al-Onaizan, Yoav Goldberg, the three anonymous reviewers and the action editor for their useful comments that improved the final version of this paper.

References

Yaser S. Abu-Mostafa. 1990. Learning from hints in neural networks. *J. Complexity*, 6(2):192–198.

Roee Aharoni and Yoav Goldberg. 2017. Towards string-to-tree neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*, pages 132–140.

Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2016. Many languages, one parser. *TACL*, 4:431–444.

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 *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2442–2452, Berlin, Germany, August. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 1957–1967.

Jonathan Baxter. 2000. A model of inductive bias learning. *JAIR*, 12:149–198.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada, July. Association for Computational Linguistics.

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 164–169, Valencia, Spain, April. Association for Computational Linguistics.

Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1455–1465, Jeju Island, Korea, July. Association for Computational Linguistics.

Ondrej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno-Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana L. Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin M. Verspoor, and Marcos Zampieri. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation, WMT 2016, colocated with ACL 2016, August 11-12, Berlin, Germany*, pages 131–198.

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The TIGER treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, volume 168.

Christian Buck, Kenneth Heafield, and Bas Van Ooyen. 2014. N-gram counts and language models from the common crawl. In *LREC*, volume 2, page 4.

Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28(1):41–75.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, volume 261, page 268.

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

Timothy Dozat and Christopher D. Manning. 2016. Deep biaffine attention for neural dependency parsing. *CoRR*, abs/1611.01734.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 199–209, San Diego, California, June. Association for Computational Linguistics.

Chris Dyer. 2017. Should neural network architecture reflect linguistic structure? In *Proceedings of the 21st Conference on Computational Natural Language*

Learning (CoNLL 2017), page 1, Vancouver, Canada, August. Association for Computational Linguistics.

Dumitru Erhan, Yoshua Bengio, Aaron C. Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. 2010. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11:625–660.

Akiko Eriguchi, Yoshimasa Tsuruoka, and Kyunghyun Cho. 2017. Learning to parse and translate improves neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 72–78, Vancouver, Canada, July. Association for Computational Linguistics.

Yoav Goldberg. 2017. Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, 10(1):1–309.

Alex Graves. 2012. *Supervised Sequence Labelling with Recurrent Neural Networks*, volume 385 of *Studies in Computational Intelligence*. Springer.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18, Boulder, Colorado, June. Association for Computational Linguistics.

Geoffrey E. Hinton and Ruslan R. Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507.

Richard A. Hudson. 1984. *Word grammar*. Blackwell Oxford.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda B. Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *TACL*, 5:339–351.

Lukasz Kaiser, Aidan N. Gomez, Noam Shazeer, Ashish Vaswani, Niki Parmar, Llion Jones, and Jakob Uszkoreit. 2017. One model to learn them all. *CoRR*, abs/1706.05137.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1700–1709.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *TACL*, 4:313–327.

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, and Noah A. Smith. 2016. Distilling an ensemble of greedy dependency parsers into one mst parser. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1744–1753, Austin, Texas, November. Association for Computational Linguistics.

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, Graham Neubig, and Noah A. Smith. 2017. What do recurrent neural network grammars learn about syntax? In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1249–1258, Valencia, Spain, April. Association for Computational Linguistics.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 228–231, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4.

Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2015a. Multi-task sequence to sequence learning. *CoRR*, abs/1511.06114.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015b. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19(2):313–330.

Héctor Martínez Alonso and Barbara Plank. 2017. When is multitask learning effective? Semantic sequence prediction under varying data conditions. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 44–53, Valencia, Spain, April. Association for Computational Linguistics.

Austin Matthews, Graham Neubig, and Chris Dyer. 2018. Using morphological knowledge in open-vocabulary neural language models. In *Meeting of the*

North American Chapter of the Association for Computational Linguistics (NAACL), New Orleans, USA, June.

Ryan T. McDonald, Koby Crammer, and Fernando C. N. Pereira. 2005. Online large-margin training of dependency parsers. In *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA*, pages 91–98.

Igor Aleksandrovič Melčuk. 1988. *Dependency Syntax: Theory and Practice*. SUNY press.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Makoto Morishita, Yusuke Oda, Graham Neubig, Koichiro Yoshino, Katsuhito Sudoh, and Satoshi Nakamura. 2017. An empirical study of minibatch creation strategies for neural machine translation. *CoRR*, abs/1706.05765.

Maria Nadejde, Siva Reddy, Rico Sennrich, Tomasz Dwojak, Marcin Junczys-Dowmunt, Philipp Koehn, and Alexandra Birch. 2017. Syntax-aware neural machine translation using CCG. *CoRR*, abs/1702.01147.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui, Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. DyNet: The dynamic neural network toolkit. *arXiv preprint arXiv:1701.03980*.

Jan Niehues and Eunah Cho. 2017. Exploiting linguistic resources for neural machine translation using multitask learning. In *Proceedings of the Second Conference on Machine Translation, WMT 2017, Copenhagen, Denmark, September 7-8, 2017*, pages 80–89.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA.*, pages 311–318.

Nils Reimers and Iryna Gurevych. 2017. Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 338–348.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 379–389.

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. In *Proceedings of the First Conference on Machine Translation, WMT 2016, colocated with ACL 2016, August 11-12, Berlin, Germany*, pages 83–91.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Anders Søgaard and Yoav Goldberg. 2016. Deep multitask learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 231–235.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey E. Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2773–2781.

Peilu Wang, Yao Qian, Frank K. Soong, Lei He, and Hai Zhao. 2015. Part-of-speech tagging with bidirectional long short-term memory recurrent neural network. *CoRR*, abs/1510.06168.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3320–3328.

Zhirui Zhang, Shujie Liu, Mu Li, Ming Zhou, and Enhong Chen. 2017. Stack-based multi-layer attention for transition-based dependency parsing. In *Proceedings of the 2017 Conference on Empirical Methods*

in Natural Language Processing, pages 1678–1683, Copenhagen, Denmark, September. Association for Computational Linguistics.

Barret Zoph and Kevin Knight. 2016. Multi-source neural translation. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 30–34.