Neural Machine Translation for Bilingually Scarce Scenarios: A Deep Multi-task Learning Approach

Poorya Zaremoodi **Gholamreza Haffari**
Faculty of Information Technology, Monash University, Australia
first.last@monash.edu

Abstract

Neural machine translation requires large amounts of parallel training text to learn a reasonable-quality translation model. This is particularly inconvenient for language pairs for which enough parallel text is not available. In this paper, we use monolingual linguistic resources in the source side to address this challenging problem based on a multi-task learning approach. More specifically, we scaffold the machine translation task on auxiliary tasks including semantic parsing, syntactic parsing, and named-entity recognition. This effectively injects semantic and/or syntactic knowledge into the translation model, which would otherwise require a large amount of training bitext. We empirically evaluate and show the effectiveness of our multi-task learning approach on three translation tasks: English-to-French, English-to-Farsi, and English-to-Vietnamese.

1 Introduction

Neural Machine Translation (NMT) with attentional encoder-decoder architectures (Luong et al., 2015; Bahdanau et al., 2015) has revolutionised machine translation, and achieved state-of-the-art for several language pairs. However, NMT is notorious for its need for large amounts of bilingual data (Koehn and Knowles, 2017) to achieve reasonable translation quality. Leveraging existing monolingual resources is a potential approach for compensating this requirement in bilingually scarce scenarios. Ideally, semantic and syntactic knowledge learned from existing linguistic resources provides NMT with proper inductive biases, leading to increased generalisation and better translation quality.

Multi-task learning (MTL) is an effective approach to inject knowledge into a task, which is learned from other related tasks. Various recent works have attempted to improve NMT with an MTL approach (Peng et al., 2017; Liu et al., 2017; Zhang and Zong, 2016); however, they either do not make use of curated linguistic resources (Domhan and Hieber, 2017; Zhang and Zong, 2016), or their MTL architectures are restrictive yielding mediocre improvements (Niehues and Cho, 2017). The current research leaves open how to best leverage curated linguistic resources in a suitable MTL framework to improve NMT.

In this paper, we make use of curated monolingual linguistic resources in the source side to improve NMT in bilingually scarce scenarios. More specifically, we scaffold the machine translation task on auxiliary tasks including semantic parsing, syntactic parsing, and named-entity recognition. This is achieved by casting the auxiliary tasks as sequence-to-sequence (SEQ2SEQ) transduction tasks, and tie the parameters of their encoders and/or decoders with those of the main translation task. Our MTL architectures makes use of deep stacked encoders and decoders, where the parameters of the top layers are shared across the tasks. We further make use of adversarial training to prevent contamination of common knowledge with task-specific information.

We present empirical results on translating from English into French, Vietnamese, and Farsi; three target languages with varying degree of divergence compared to English. Our extensive empirical results demonstrate the effectiveness of our MTL approach in substantially improving the translation quality for these three translation tasks in bilingually scarce scenarios.

2 Neural SEQ2SEQ Transduction

Our MTL is based on the attentional encoder-decoder architecture for SEQ2SEQ transduction. It contains an encoder to *read* the input sentence, and an attentional decoder to *generate* the output.

Encoder The encoder is a bi-directional RNN whose hidden states represent tokens of the input sequence. These representations capture information not only of the corresponding token, but also other tokens in the sequence to leverage the context. The bi-directional RNN consists of two RNNs running in the left-to-right and right-to-left directions over the input sequence:

$$\begin{aligned}\overrightarrow{\boldsymbol{h}}_i &= \mathrm{RNN}(\overrightarrow{\boldsymbol{h}}_{i-1}, \boldsymbol{E}_S[x_i]) \\ \overleftarrow{\boldsymbol{h}}_i &= \mathrm{RNN}(\overleftarrow{\boldsymbol{h}}_{i+1}, \boldsymbol{E}_S[x_i])\end{aligned}$$

where $\boldsymbol{E}_S[x_i]$ is the embedding of the token x_i from the embedding table $\boldsymbol{E}_S$ of the input (source) space, and $\overrightarrow{\boldsymbol{h}}_i$ and $\overleftarrow{\boldsymbol{h}}_i$ are the hidden states of the forward and backward RNNs which can be based on the LSTM (long-short term memory) (Hochreiter and Schmidhuber, 1997) or GRU (gated-recurrent unit) (Chung et al., 2014) units. Each source token is then represented by the concatenation of the corresponding bidirectional hidden states, $\boldsymbol{h}_i = [\overrightarrow{\boldsymbol{h}}_i; \overleftarrow{\boldsymbol{h}}_i]$.

Decoder. The backbone of the decoder is a unidirectional RNN which generates the token of the output one-by-one from left to right. The generation of each token y_j is conditioned on all of the previously generated tokens $\boldsymbol{y}_{<j}$ via the state of the RNN decoder $\boldsymbol{s}_j$, and the input sequence via a *dynamic* context vector $\boldsymbol{c}_j$ (explained shortly):

$$\begin{aligned}y_j &\sim \mathrm{softmax}(\boldsymbol{W}_y \cdot \boldsymbol{r}_j + \boldsymbol{b}_r) \quad (1)\\ \boldsymbol{r}_j &= \tanh(\boldsymbol{s}_j + \boldsymbol{W}_{rc} \cdot \boldsymbol{c}_j + \boldsymbol{W}_{rj} \cdot \boldsymbol{E}_T[y_{j-1}]) \quad (2)\\ \boldsymbol{s}_j &= \tanh(\boldsymbol{W}_s \cdot \boldsymbol{s}_{j-1} + \boldsymbol{W}_{sj} \cdot \boldsymbol{E}_T[y_{j-1}] + \boldsymbol{W}_{sc} \cdot \boldsymbol{c}_j)\end{aligned}$$

where $\boldsymbol{E}_T[y_j]$ is the embedding of the token y_j from the embedding table $\boldsymbol{E}_T$ of the output (target) space, and the $\boldsymbol{W}$ matrices and $\boldsymbol{b}_r$ vector are the parameters.

A crucial element of the decoder is the *attention* mechanism which dynamically attends to relevant parts of the input sequence necessary for generating the next token in the output sequence. Before generating the next token t_j, the decoder computes the attention vector $\boldsymbol{\alpha}_j$ over the input token:

$$\begin{aligned}\boldsymbol{\alpha}_j &= \mathrm{softmax}(\boldsymbol{a}_j)\\ a_{ji} &= \boldsymbol{v} \cdot \tanh(\boldsymbol{W}_{ae} \cdot \boldsymbol{h}_i + \mathbf{W}_{at} \cdot \boldsymbol{s}_{j-1})\end{aligned}$$

which intuitively is similar to the notion of *alignment* in word/phrase-based statistical MT (Brown et al., 1993). The attention vector is then used to compute a fixed-length dynamic representation of the source sentence

$$\boldsymbol{c}_j = \sum_i \alpha_{ji} \boldsymbol{h}_i. \quad (3)$$

which is conditioned upon in the RNN decoder when computing the next state or generating the output word (as mentioned above).

Training and Decoding. The model parameters are trained end-to-end by maximising the (regularised) log-likelihood of the training data

$$\arg\max_{\boldsymbol{\theta}} \sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}} \sum_{j=1}^{|\boldsymbol{y}|} \log P_{\boldsymbol{\theta}}(y_j|\boldsymbol{y}_{<j}, \boldsymbol{x})$$

where the above conditional probability is defined according to eqn (1). Usually drop-out is employed to prevent over-fitting on the training data. In the decoding time, the best output sequence for a given input sequence is produced by

$$\arg\max_{\boldsymbol{y}} P_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x}) = \prod_j P_{\boldsymbol{\theta}}(y_j|\boldsymbol{y}_{<j}\boldsymbol{x}).$$

Usually greedy decoding or beam search algorithms are employed to find an approximate solution, since solving the above optimisation problem exactly is computationally hard.

3 SEQ2SEQ Multi-Task Learning

We consider an extension of the basic SEQ2SEQ model where the encoder and decoder are equipped with deep stacked layers. Presumably, deeper layers capture more abstract information about a task, hence they can be used as a mechanism to share useful generalisable information among multiple tasks.

Deep Stacked Encoder. The deep encoder consists of multiple layers, where the hidden states in layer $\ell-1$ are the inputs to the hidden states at the next layer ℓ. That is,

$$\begin{aligned}\overrightarrow{\boldsymbol{h}}^\ell_i &= \overrightarrow{\mathrm{RNN}}^\ell_{\boldsymbol{\theta}_{\ell,enc}}(\overrightarrow{\boldsymbol{h}}^\ell_{i-1}, \boldsymbol{h}^{\ell-1}_i)\\ \overleftarrow{\boldsymbol{h}}^\ell_i &= \overleftarrow{\mathrm{RNN}}^\ell_{\boldsymbol{\theta}_{\ell,enc}}(\overleftarrow{\boldsymbol{h}}^\ell_{i-1}, \boldsymbol{h}^{\ell-1}_i)\end{aligned}$$

where $\boldsymbol{h}^\ell_i = [\overrightarrow{\boldsymbol{h}}^\ell_i; \overleftarrow{\boldsymbol{h}}^\ell_i]$ is the hidden state of the ℓ'th layer RNN encoder for the i'th source sentence word. The inputs to the first layer forward/backward RNNs are the source word embeddings $\boldsymbol{E}_S[x_i]$. The representation of the source sentence is then the concatenation of the hidden states for all layers $\boldsymbol{h}_i = [\boldsymbol{h}^1_i; \ldots; \boldsymbol{h}^L_i]$ which is then used by the decoder.

Deep Stacked Decoder. Similar to the multi-layer RNN encoder, the decoder RNN has multiple layers:

$$\boldsymbol{s}_j^{\ell} = \mathrm{RNN}^{\ell}_{\boldsymbol{\theta}_{\ell,dec}}(\boldsymbol{s}_{j-1}^{\ell}, \boldsymbol{s}_j^{\ell-1})$$

where the inputs to the first layer RNNs are

$$\boldsymbol{W}_{sj} \cdot \boldsymbol{E}_T[y_{j-1}] + \boldsymbol{W}_{sc} \cdot \boldsymbol{c}_j$$

in which $\boldsymbol{c}_j$ is the dynamic source context, as defined in eqn 3. The state of the decoder is then the concatenation of the hidden states for all layers: $\boldsymbol{s}_j = [\boldsymbol{s}_j^1; \ldots; \boldsymbol{s}_j^L]$ which is then used in eqn 2 as part of the "output generation module".

Shared Layer MTL. We share the deep layer RNNs in the encoders and/or decoders across the tasks, as a mechanism to share abstract knowledge and increase model generalisation.

Suppose we have a total of $M + 1$ tasks, consisting of the main task plus M auxiliary tasks. Let $\boldsymbol{\Theta}^m_{enc} = \{\boldsymbol{\theta}^m_{\ell,enc}\}_{\ell=1}^{L}$ and $\boldsymbol{\Theta}^m_{dec} = \{\boldsymbol{\theta}^m_{\ell',dec}\}_{\ell'=1}^{L'}$ be the parameters of multi-layer encoder and decoder for the task m. Let $\{\boldsymbol{\Theta}^m_{enc}, \boldsymbol{\Theta}^m_{dec}\}_{m=1}^{M}$ and $\{\boldsymbol{\Theta}^0_{enc}, \boldsymbol{\Theta}^0_{dec}\}$ be the RNN parameters for the auxiliary tasks and the main task, respectively. We share the parameters of the deep-level encoders and decoders of the auxiliary tasks with those of the main task. That is,

$$\begin{aligned}
\forall m \in [1,..,M]\ \forall \ell \in [L^m_{enc},..,L] \quad &: \quad \boldsymbol{\theta}^m_{\ell,enc} = \boldsymbol{\theta}^0_{\ell,enc} \\
\forall m \in [1,..,M]\ \forall \ell' \in [L'^m_{dec},..,L'] \quad &: \quad \boldsymbol{\theta}^m_{\ell',dec} = \boldsymbol{\theta}^0_{\ell',dec}
\end{aligned}$$

where L^m_{enc} and L'^m_{dec} specify the deep-layer RNNs need to be shared parameters. Other parameters to share across the tasks include those of the attention module, the source/target embedding tables, and the output generation module. As an extreme case, we can share *all* the parameters of SEQ2SEQ architectures across the tasks.

Training Objective. Suppose we are given a collection of $M+1$ SEQ2SEQ transductions tasks, each of which is associated with a training set $\mathcal{D}_m := \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N_m}$. The parameters are learned by maximising the MTL training objective:

$$\mathcal{L}_{mtl}(\boldsymbol{\Theta}_{mtl}) := \sum_{m=0}^{M} \frac{\gamma_m}{|\mathcal{D}_m|} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{D}_m} \log P_{\boldsymbol{\Theta}_m}(\boldsymbol{y}|\boldsymbol{x}) \tag{4}$$

where $\boldsymbol{\Theta}_{mtl}$ denotes all the parameters of the MTL architecture, $|\mathcal{D}_m|$ denotes the size of the training set for the task m, and γ_m balances out its influence in the training objective.

Training Schedule. Variants of stochastic gradient descent (SGD) can be used to optimise the objective in order to learn the parameters. Making the best use of tasks with different objective geometries is challenging, e.g. due to the scale of their gradients. One strategy for making an SGD update is to select the tasks from which the next data items should be chosen. In our training schedule, we randomly select a training data item from the main task, and pair it with a data item selected from a randomly selected auxiliary task for making the next SGD update. This ensures the presence of training signal from the main task in all SGD updates, and avoids the training signal being washed out by the auxiliary tasks.

4 Adversarial Training

The learned shared knowledge can be contaminated by task-specific information. We address this issue by adding an adversarial objective. The basic idea is to augment the MTL training objective with additional terms, so that the identity of a task cannot be predicted from its data items by the representations resulted from the shared encoder/decoder RNN layers.

Task Discriminator. The goal of the task discriminator is to predict the identity of a task for a data item based on the representations of the share layers. More specifically, our task discriminator consists of two RNNs with LSTM units, each of which encodes the sequence of hidden states in the shared layers of the encoder and the decoder.[1] The last hidden states of these two RNNs are then concatenated, giving rise to a fixed dimensional vector summarising the representations in the shared layers. The summary vector is passed through a fully connected layer followed by a $\mathrm{softmax}$ to predict the probability distribution over the tasks:

$$\begin{aligned}
P_{\boldsymbol{\Theta}_d}(\text{task id}|\boldsymbol{h}_d) &\sim \mathrm{softmax}(\boldsymbol{W}_d \boldsymbol{h}_d + \boldsymbol{b}_d) \\
\boldsymbol{h}_d &:= \mathrm{disLSTMs}(\mathrm{shrRep}_{\boldsymbol{\Theta}_{mtl}}(\boldsymbol{x}, \boldsymbol{y}))
\end{aligned}$$

where $\mathrm{disLSTMs}$ denotes the discriminator LSTMs, $\mathrm{shrRep}_{\boldsymbol{\Theta}_{mtl}}(\boldsymbol{x}, \boldsymbol{y})$ denotes the representations in the shared layer of deep encoders and decoders in the MTL architecture, and $\boldsymbol{\Theta}_d$ includes the $\mathrm{disLSTMs}$ parameters as well as $\{\boldsymbol{W}_d, \boldsymbol{b}_d\}$.

[1] When multiple layers are shared, we concatenate their hidden states at each time step, which is then input to the task discriminator's LSTMs.

Adversarial Objective. Inspired by (Chen et al., 2017), we add two additional terms to the MTL training objective in eqn 4. The first term is $\mathcal{L}_{adv1}(\boldsymbol{\Theta}_d)$ defined as:

$$\sum_{m=0}^{M} \sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}_m} \log P_{\boldsymbol{\Theta}_d}(m|\,\text{disLSTMs}(\text{shrRep}_{\boldsymbol{\Theta}_{mtl}}(\boldsymbol{x},\boldsymbol{y}))).$$

Maximising the above objective over $\boldsymbol{\Theta}_d$ ensures proper training of the discriminator to predict the identity of the task. The second term ensures that the parameters of the shared layers are trained so that they confuse the discriminator by maximising the entropy of its predicted distribution over the task identities. That is, we add the term $\mathcal{L}_{adv2}(\boldsymbol{\Theta}_{mtl})$ to the training objective defined as:

$$\sum_{m=0}^{M} \sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}_m} H\left[P_{\boldsymbol{\Theta}_d}(.|\,\text{disLSTMs}(\text{shrRep}_{\boldsymbol{\Theta}_{mtl}}(\boldsymbol{x},\boldsymbol{y})))\right]$$

where $H[.]$ is the entropy of a distribution. In summary, the adversarial training leads to the following optimisation

$$\arg\max_{\boldsymbol{\Theta}_d,\boldsymbol{\Theta}_{mtl}} \mathcal{L}_{mtl}(\boldsymbol{\Theta}_{mtl})+\mathcal{L}_{adv1}(\boldsymbol{\Theta}_d)+\lambda\mathcal{L}_{adv2}(\boldsymbol{\Theta}_{mtl}).$$

We maximise the above objective by SGD, and update the parameters by alternating between optimising $\mathcal{L}_{mtl}(\boldsymbol{\Theta}_{mtl}) + \lambda\mathcal{L}_{adv2}(\boldsymbol{\Theta}_{mtl})$ and $\mathcal{L}_{adv1}(\boldsymbol{\Theta}_d)$.

5 Experiments

5.1 Bilingual Corpora

We use three language-pairs, translating from English to French, Farsi, and Vietnamese. We have chosen these languages to analyse the effect of multi-task learning on languages with different underlying linguistic structures. The sentences are segmented using BPE (Sennrich et al., 2016) on the union of source and target vocabularies for English-French and English-Vietnamese. For English-Farsi, BPE is performed using separate vocabularies due to the disjoint alphabets. We use a special <UNK> token to replace unknown BPE units in the test and development sets.

Table 1 show some statistics about the bilingual corpora. Further details about the corpora and their pre-processing is as follows:

- The English-French corpus is a random subset of EuroParlv7 as distributed to WMT2014. Sentence pairs in which either the source or the target has length more than 80 (before applying BPE) have been removed. The BPE is performed with a 30k total vocabulary size. The “news-test2012” and “news-test-2013” portions are used for validation and test sets, respectively.

	Train	Dev	Test
En → Fr	98,846	5,357	5,357
En → Fa	98,158	3,000	4,000
En → vi	133,290	1,553	1,268

Table 1: The statistics of bilingual corpora.

- The English-Farsi corpus is assembled from all the parallel news text in LDC2016E93 *Farsi Representative Language Pack* from the Linguistic Data Consortium, combined with English-Farsi parallel subtitles from the TED corpus (Tiedemann, 2012). Since the TED subtitles are user-contributed, this text contained considerable variation in the encoding of its Perso-Arabic characters. To address this issue, we have normalized the corpus using the Hazm toolkit[2]. Sentence pairs in which one of the sentences has more than 80 (before applying BPE) are removed, and BPE is performed with a 30k vocabulary size. Random subsets of this corpus (3k and 4k sentences each) are held out as validation and test sets, respectively.

- The English-Vietnamese is from the translation task in IWSLT 2015, and we use the preprocessed version provided by (Luong and Manning, 2015). The sentence pairs in which at least one of their sentences had more than 300 units (after applying BPE) are removed. “tst2012” and “tst2013” parts are used for validation and test sets, respectively.

5.2 Auxiliary Tasks

We have chosen the following auxiliary tasks to provide the NMT model with syntactic and/or semantic knowledge, in order to enhance the quality of translation:

Named-Entity Recognition (NER). With a small bilingual training corpus, it would be hard for the NMT model to learn how to translate rarely occurring named-entities. Through the NER task,

[2] www.sobhe.ir/hazm

the model hopefully learns the skill to recognize named entities. Speculatively, it would then enables leaning translation patterns by masking out named entities. The NER data comes from the CONLL shared task.[3]

Syntactic Parsing. This task enables NMT to learn the phrase structure of the input sentence, which would then be useful in better re-orderings. This would be most useful for language pairs with high syntactic divergence. The parsing data comes from the Penn Tree Bank with the standard split for training, development, and test (Marcus et al., 1993). We linearise the constituency trees, in order to turn syntactic parsing as a SEQ2SEQ transduction (Vinyals et al., 2015).

Semantic Parsing. A good translation should preserve the meaning. Learning from the semantic parsing task enables the NMT model to pay attention to a meaning abstraction of the source sentence, in order to convey it to the target translation. We have made use of the Abstract Meaning Representation (AMR) corpus Release 2.0 (LDC2017T10), which pairs English sentences AMR meaning graphs. We linearise the AMR graphs, in order to convert semantic parsing as a SEQ2SEQ transduction problem (Konstas et al., 2017).

5.3 Models and Baselines

We have implemented the proposed multi-task learning architecture in C++ using DyNet (Neubig et al., 2017), on top of Mantis (Cohn et al., 2016) which is an implementation of the attentional SEQ2SEQ NMT model in (**?**). In our multitask architecture, we do partial sharing of parameters, where the parameters of the top 2 stacked layers are shared among the encoders of the tasks. Moreover, we share the parameters of the top layer stacked decoder among the tasks. Source and target embedding tables are shared among the tasks, while the attention component is task-specific. [4] We compare against the following baselines:

- Baseline 1: The vanila SEQ2SEQ model without any multi-tasking.
- Baseline 2: The multi-tasking architecture proposed in (Niehues and Cho, 2017), which is a special case of our approach where the parameters of all 3 stacked layers are shared among the tasks.[5] They have not used deep stacked layers in encoder and decoder as we do, so we extend their work to make it comparable with ours.

The configuration of models is as follows. The encoders and decoders make use of GRU units with 400 hidden dimensions, and the attention component has 200 dimensions. For training, we used Adam algorithm (Kingma and Ba, 2014) with the initial learning rate of 0.003 for all of the tasks. Learning rates are halved when the performance on the corresponding dev set decreased. In order to speed-up the training, we use mini-batching with the size of 32. Dropout rates for both encoder and decoder are set to 0.5, and models are trained for 50 epochs where the best models is selected based on the perplexity on the dev set. λ for the adversarial training is set to 0.5. Once trained, the NMT model translates using the greedy search. We use BLEU (Papineni et al., 2002) to measure translation quality. [6]

5.4 Results

Table 2 reports the BLEU scores and perplexities for the baseline and our proposed method on the three aforementioned translation tasks. It can be seen that the performance of multi-task learning models are better than Baseline 1 (only MT task). This confirms that adding auxiliary tasks helps to increase the performance of the machine translation task.

As expected, the effect of different tasks are not similar across the language pairs, possibly due to the following reasons: (i) these translation tasks datasets come from different domains so they have various degree of domain relatedness to the auxiliary tasks, and (ii) the BLEU scores of the Baseline 1 show that the three translation models are on different quality levels which may entail that they benefit from auxiliary knowledge on different levels. In order to improve a model with low quality translations due to language divergence, syntactic knowledge can be more helpful as they help better reorderings. In a higher-quality model, however, semantic knowledge can be more useful as

[3] https://www.clips.uantwerpen.be/conll2003/ner

[4] In our experiments, models with task-specific attention components achieved better results than those sharing them.

[5] We have used their best performing architecture and changed the training schedule to ours.

[6] With "multi-bleu.perl" script from Moses (Koehn et al., 2007).

	English → French				English → Farsi				English → Vietnamese			
	Dev		Test		Dev		Test		Dev		Test	
	PPL	BLEU	PPL	BLEU	PPL	BLEU	PPL	BLEU	PPL	BLEU	PPL	BLEU
NMT	117.27	8.85	80.29	10.71	86.63	7.69	87.94	7.46	23.24	16.53	20.36	17.86
+ Semantic	71.7	10.58	51.2	12.72	56.32	8.3	57.88	8.32	14.86	19.96	12.79	21.82
+ NER	73.42	10.73	52.07	12.92	48.46	9.11	49.53	9.03	15.04	20.2	13.13	21.96
+ Syntactic	69.45	11.88	48.9	13.94	44.35	9.73	45.37	9.37	16.42	18.4	14.27	20.4
+ All Tasks	69.71	11.3	49.86	13.41	44.03	**9.68**	45.1	**9.7**	14.79	20.12	12.65	22.41
+ All+Adv.	68.44	**11.93**	48.92	**14.02**	45.25	9.55	45.87	9.19	14.19	**21.21**	12.11	**23.54**

Table 2: BLEU scores and perplexities of the baseline vs our MTL architecture with various auxiliary tasks on the full bilingual datasets.

	W/O Adaptation		W/ Adaptation		
	Partial	Full	Partial	Part.+Adv.	Full
En→Fr	13.41	9.94	14.86	15.12	11.94
En → Fa	9.7	7.89	10.31	10.08	8.6
En → Vi	22.41	20.26	23.35	24.28	21.67

Table 3: Our method (partial parameter sharing) against Baseline 2 (full parameter sharing).

a higher-level linguistic knowledge. This pattern can be seen in the reported results: syntactic parsing leads to more improvement on Farsi translation which has a low BLEU score and high language divergence to English, and semantic parsing yields more improvement on the Vietnamese translation task which already has a high BLEU score. The NER task has led to a steady improvement in all the translation tasks, as it leads to better handling of named entities.

We have further added adversarial training to ensure the shared representation learned by the encoder is not contaminated by the task-specific information. The results are in the last row of Table 2. The experiments show that adversarial training leads to further gains in MTL translation quality, except when translating into Farsi. We speculate this is due to the low quality of NMT for Farsi, where updating shared parameters with respect to the entropy of discriminator's predicted distribution may negatively affect the model.

Table 3 compares our multi-task learning approach to Baseline 2. As Table 3, our partial parameter sharing mechanism is more effective than fully sharing the parameters (Baseline 2), due to its flexibility in allowing access to private task-specific knowledge. We also applied the adaptation technique (Niehues and Cho, 2017) as follows. Upon finishing MTL training, we continue to train only on the MT task for another 20 epochs, and choose the best model based on perplexity on dev set. Adaptation has led to consistent gains in the performance of our MTL architecture and Baseline 2.

5.5 Analysis

How many layers of encoder/decoder to share? Figure 2 show the results of changing the number of shared layers in encoder and decoder based on the En→Vi translation task. The results confirm that partial sharing of stacked layers is better than full sharing. Intuitively, partial sharing provides the model with an opportunity to learn task specific skills via the private layers, while leveraging the knowledge learned from other tasks via shared layers.

Statistics of gold n-grams in MTL translations. Generating high order gold n-grams is hard. We analyse the effect of syntactic and semantic knowledge on generating gold n-grams in translations.

For each sentence, we first extract n-grams in the gold translation, and then compute the number of n-grams which are common with the generated translations. Finally, after aggregating the results over the entire test set, we compute the percentage of additional gold n-grams generated by each MTL model compared to the ones in single-task MT model. The results are depicted in Figure 1. Interestingly, the MTL models generate more correct n-grams relative to the vanilla NMT model, as n increases.

Effect of the NER task. The NMT model has difficulty translating rarely occurring named-entities, particularly when the bilingual parallel data is scarce. We expect learning from the NER task leads the MTL model to recognize named-entities and learn underlying patterns for translating them. The top part in Table 4 shows an example of such situation. As seen, the MTL is able to recognize all of the named-entities in the sentence and translate the while the single-task model

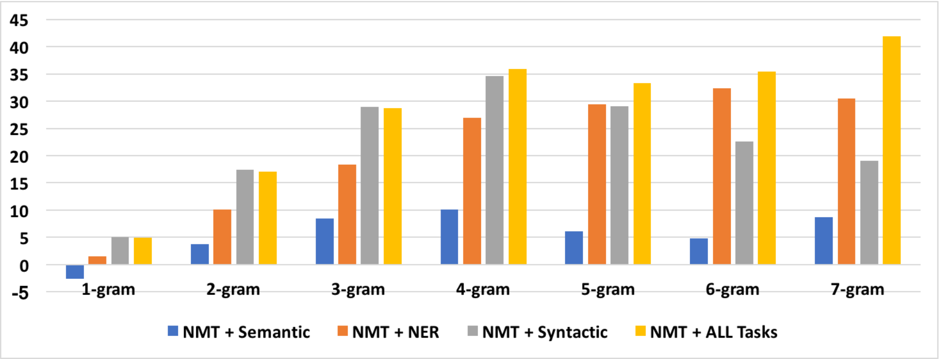

Figure 1: Percentage of more correct n-grams generated by the deep MTL models compared to the single-task model (only MT).

English	this is a building in Hyderabad , India .
Reference	this a building in Hyderabad is , in India .
MT only model	this a building in Hyderabad is .
MT+NER model	this a building in Hyderabad India is .
English	we see people on our screens .
Reference	we people on television screen or cinema see .
MT only model	we people see we people .
MT+semantic model	we people on television screen see .
English	in hospitals , for new medical instruments ; in streets for traffic control .
Reference	in hospitals , for instruments medical new ; in streets for control traffic
MT only model	in hospitals , for tools new tools for traffic controlled*[7] .
MT+syntactic model	in hospitals , for devices new , in streets for control traffic .

Table 4: Example of translations on Farsi test set. In this examples each Farsi word is replaced with its English translation, and the order of words is reversed (Farsi is written right-to-left). The structure of Farsi is Subject-Object-Verb (SOV), leading to different word orders in English and Reference sentences.

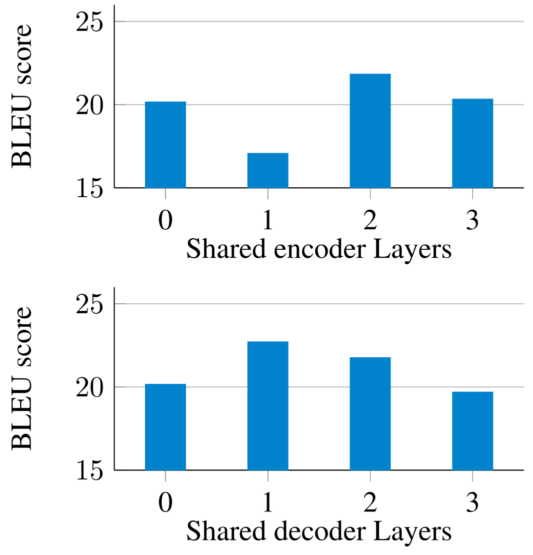

Figure 2: BLEU scores for different numbers of shared layers in (top) encoder while no layer is shared in decoder, and (bottom) decoder while no layer is shared in encoder

missed "India".

For more analysis, we have applied a Farsi POS tagger (Feely et al., 2014) to gold translations. Then, we extracted n-grams with at least one noun in them, and report the statistics of correct such n-grams, similar to what reported in Figure 1. The resulting statistics is depicted in Figure 3. As seen, the MTL model trained on MT and NER tasks leads to generation of more correct unigram noun phrases relative to the vanilla NMT, as n increases.

Effect of the semantic parsing task. Semantic parsing encourages a precise understanding of the source text, which would then be useful for conveying the correct meaning to the translation. The middle part in Table 4 is an example translation, showing that semantic parsing has helped NMT by understanding that "the subject sees the object via subject's screens".

Effect of the syntactic parsing task. Recognizing the syntactic structure of the source sentence helps NMT to better translate phrases. The bottom part of Table 4 shows an example translation demonstrating such case. The source sentence is talking about "a method for controlling the

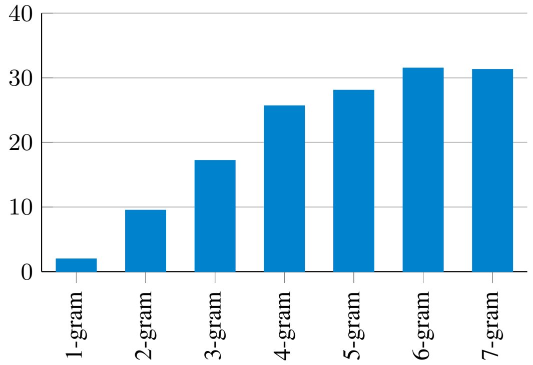

Figure 3: Percentage of more corrected n-grams with at least one noun generated by MT+NER model compared with the only MT model (only MT).

traffic", which is correctly translated by the MTL model while vanilla NMT has mistakenly translated it to "controlled traffic".

6 Related Work

Multi-task learning has attracted attention to improve NMT in recent work. (Zhang and Zong, 2016) has made use of monolingual data in the source language in a multitask learning framework by sharing encoder in the attentional encoder-decoder model. Their auxiliary task is to reorder the source text to make it close to the target language word order. (Domhan and Hieber, 2017) proposed a two-layer stacked decoder, which the bottom layer is trained on language modelling on the target language text. The next word is jointly predicted by the bottom layer language model and the top layer attentional RNN decoder. They reported only moderate improvements over the baseline and fall short against using synthetic parallel data. (Dalvi et al., 2017) investigated the amount of learned morphology and how it can be injected using MTL. Our method is related to what they call joint data-learning, where they share all of the SEQ2SEQ components among the tasks.

(Belinkov et al., 2017a; Shi et al., 2016; Belinkov et al., 2017b) investigate syntax/semantics phenomena learned as a byproduct of SEQ2SEQ NMT training. We, in turn, investigate the effect of injecting syntax/semantic on learning NMT using MTL.

The closet work to ours is (Niehues and Cho, 2017), which has made use of part-of-speech tagging and named-entity recognition tasks to improve NMT. They have used the attentional encoder-decoder with a shallow architecture, and share different parts eg the encoder, decoder, and attention. They report the best performance with fully sharing the encoder. In contrast, our architecture uses partial sharing on deep stacked encoder and decoder components, and the results show that it is critical for NMT improvement in MTL. Furthermore, we propose adversarial training to prevent contamination of shared knowledge with task specific details.

Taking another approach to MTL, (Søgaard and Goldberg, 2016) and (Hashimoto et al., 2017) have proposed architectures by stacking up tasks on top of each other according to their linguistic level, eg from lower level tasks (POS tagging) to higher level tasks (parsing). In this approach, each task uses predicted annotations and hidden states of the lower-level tasks for making a better prediction. This is contrast to the approach taken in this paper where models with shared parameters are trained jointly on multiple tasks.

More broadly, deep multitask learning has been used for various NLP problems, including graph-based parsing (Chen and Ye, 2011) and keyphrase boundary classification (Augenstein and Søgaard, 2017) . (Chen et al., 2017) has applied multi-task learning for Chinese word segmentation, and (Liu et al., 2017) applied it for text classification problem. Both of these works have used adversarial training to make sure the shared layer extract only common knowledge.

MTL has been used effectively to learn from multimodal data. (Luong et al., 2016) has proposed MTL architectures for neural SEQ2SEQ transduction for tasks including MT, image caption generation, and parsing. They fully share the encoders (many-to-one), the decoders (one-to-many), or some of the encoders and decoders (many-to-many). (Pasunuru and Bansal, 2017) have made use of an MTL approach to improve video captioning with auxiliary tasks including video prediction and logical language entailment based on a many-to-many architecture.

7 Conclusions and Future Work

We have presented an approach to improve NMT in bilingually scarce scenarios, by leveraging curated linguistic resources in the source, including semantic parsing, syntactic parsing, and named entity recognition. This is achieved via an effective MTL architecture, based on deep stacked encoders and decoders, to share common knowledge among the MT and auxiliary tasks. Our experimental results show substantial improvements in the translation quality, when translating from English to French, Vietnamese, and Farsi in bilingually scarce scenarios. For future work, we would like to investigate architectures which allow automatic parameter tying among the tasks (Ruder et al., 2017).

Acknowledgments

The research reported here was initiated at the 2017 Frederick Jelinek Memorial Summer Workshop on Speech and Language Technologies, hosted at Carnegie Mellon University and sponsored by Johns Hopkins University with unrestricted gifts from Amazon, Apple, Facebook, Google, and Microsoft. We are very grateful to the workshop members for the insightful discussions and data pre-processing. This work was supported by computational resources from the Multi-modal Australian ScienceS Imaging and Visualisation Environment (MASSIVE) at Monash University. This work was supported by the Australian Research Council via DP160102686. The first author was partly supported by CSIRO's Data61.

References

Isabelle Augenstein and Anders Søgaard. 2017. Multitask learning of keyphrase boundary classification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. pages 341–346.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of the International Conference on Learning Representations*.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017a. What do neural machine translation models learn about morphology? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. pages 861–872.

Yonatan Belinkov, Lluís Màrquez, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2017b. Evaluating layers of representation in neural machine translation on part-of-speech and semantic tagging tasks. In *Proceedings of the International Joint Conference on Natural Language Processing*. pages 1–10.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics* 19(2):263–311.

Xinchi Chen, Zhan Shi, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-criteria learning for chinese word segmentation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. pages 1193–1203.

Y. Chen and X. Ye. 2011. Projection Onto A Simplex . *arXiv preprint arXiv:1101.6081* .

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. NIPS Workshop on Deep Learning.

Trevor Cohn, Cong Duy Vu Hoang, Ekaterina Vymolova, Kaisheng Yao, Chris Dyer, and Gholamreza Haffari. 2016. Incorporating Structural Alignment Biases into an Attentional Neural Translation Model. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics–Human Language Technologies*. pages 876–885.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, Yonatan Belinkov, and Stephan Vogel. 2017. Understanding and improving morphological learning in the neural machine translation decoder. In *Proceedings of the International Joint Conference on Natural Language Processing*. pages 142–151.

Tobias Domhan and Felix Hieber. 2017. Using target-side monolingual data for neural machine translation through multi-task learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pages 1501–1506.

Weston Feely, Mehdi Manshadi, Robert E Frederking, and Lori S Levin. 2014. The CMU METAL Farsi NLP Approach. In *LREC*. pages 4052–4055.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pages 1923–1933.

Sepp Hochreiter and Jurgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9(8):1735–1780.

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

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the Annual Meeting of the ACL* *on Interactive Poster and Demonstration Sessions*. pages 177–180.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*. pages 28–39.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural amr: Sequence-to-sequence models for parsing and generation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. pages 146–157.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-task learning for text classification. *arXiv preprint arXiv:1704.05742* .

Minh-Thang Luong and Christopher D. Manning. 2015. Stanford neural machine translation systems for spoken language domain. In *International Workshop on Spoken Language Translation*. Da Nang, Vietnam.

Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In *International Conference on Learning Representations*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1412–1421.

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

G. Neubig, C. Dyer, Y. Goldberg, A. Matthews, W. Ammar, A. Anastasopoulos, M. Ballesteros, D. Chiang, D. Clothiaux, T. Cohn, K. Duh, M. Faruqui, C. Gan, D. Garrette, Y. Ji, L. Kong, A. Kuncoro, G. Kumar, C. Malaviya, P. Michel, Y. Oda, M. Richardson, N. Saphra, S. Swayamdipta, and P. Yin. 2017. DyNet: The Dynamic Neural Network Toolkit. *ArXiv preprints arXiv:1701.03980* .

Jan Niehues and Eunah Cho. 2017. Exploiting linguistic resources for neural machine translation using multi-task learning. In *Proceedings of the Second Conference on Machine Translation*. 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 Annual Meeting on Association for Computational Linguistics*. pages 311–318.

Ramakanth Pasunuru and Mohit Bansal. 2017. Multi-task video captioning with video and entailment generation. In *Proceedings of ACL*.

Hao Peng, Sam Thomson, and Noah A Smith. 2017. Deep multitask learning for semantic dependency parsing. *arXiv preprint arXiv:1704.06855* .

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2017. Sluice networks: Learning what to share between loosely related tasks. *arXiv preprint arXiv:1705.08142* .

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. pages 1715–1725.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural mt learn source syntax? In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pages 1526–1534.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. pages 231–235.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *Proceedings of the International Conference on Language Resources and Evaluation*. pages 2214–2218.

Oriol Vinyals, Ł ukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems*, pages 2773–2781.

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pages 1535–1545.