KIT's Multilingual Neural Machine Translation systems for IWSLT 2017

†*Ngoc-Quan Pham,* †*Matthias Sperber,* *†*Elizabeth Salesky,* †*Thanh-Le Ha,* †*Jan Niehues,* *†*Alexander Waibel*

†Institute for Anthropomatics and Robotics
KIT - Karlsruhe Institute of Technology, Germany
firstname.lastname@kit.edu
* Language Technologies Institute
Carnegie Mellon University
firstname.lastname@cmu.edu

Abstract

In this paper, we present KIT's multilingual neural machine translation (NMT) systems for the IWSLT 2017 evaluation campaign machine translation (MT) and spoken language translation (SLT) tasks.

For our MT task submissions, we used our multi-task system, modified from a standard attentional neural machine translation framework, instead of building 20 individual NMT systems. We investigated different architectures as well as different data corpora in training such a multilingual system. We also suggested an effective adaptation scheme for multilingual systems which brings great improvements compared to monolingual systems.

For the SLT track, in addition to a monolingual neural translation system used to generate correct punctuations and true cases of the data prior to training our multilingual system, we introduced a noise model in order to make our system more robust. Results show that our novel modifications improved our systems considerably on all tasks.

1. Introduction

In recent works, attention-based neural networks has been considered the state-of-the-art approach for machine translation. More importantly, this framework can be efficiently adapted or customized to fit in a multilingual setting, so that one model can be trained to translate from and to multiple languages. In this evaluation campaign, we empirically explore different architectures which have been exploited in various previous works, in order to find the best combination for the multilingual setting.

Specifically, we break down the neural machine translation architecture into its main components: embedding layers, encoders, decoders, attention and output layers. Our analysis indicates which components can be shared to benefit from multilingual data. We also employed an adaptation strategy which is proved to be beneficial for multi-task learning. Our best systems are the ensembles of individual architectures.

2. Data Processing

The data is preprocessed prior to training and translation. Sentence longer than 50 words and aligned sentence pairs having a big difference in length are removed. Special dates, numbers and symbols are normalized. Smartcasing is applied as well. Afterwards, we apply byte pair encoding [1] to model the translation of rare words. We build corpora using 40K codes and 80K codes. Since we did not see a large difference in performance, all reported results use a byte pair encoding size of 40K.

2.1. Sentence alignment

While for the in-domain TED corpus, parallel data was provided for all directions, the out-of-domain EPPS data was only available from and to English. For all language direction that do not include English, no additional data was available. In order to generate this data, we used English as a pivot language, sentence-aligning the English sides of source-English and English-target data in order to extract source-target sentence pairs. In the two tracks that we participated in, the Small data consists of 4.2 million sentences while the Large data has 26 million for 20 language directions.

3. Multilingual NMT

In previous works, the encoder-decoder architecture with attention mechanism has been used in a multilingual setting [2, 3, 4, 5] with various architectural choices. While most authors decided to share the encoder and decoder weights between languages, the attention module remains controversial, as [2] negates the use of attention in the multi-task models, [4] uses explicit attention layers for each language pair, and in [3], one single model is shared between all pairs. In this work, we explore the possibilities of architectural sharing between encoder, decoder and attention layers across languages.

3.1. Architectures

3.1.1. Neural Machine Translation

Our base model is the encoder-decoder with attention mechanism [6, 7], in which both of the encoder and the decoder are Long-Short Term Memory networks [8]. The attention module is a two-layer feed-forward neural network that we found to work better than simple dot-product or bilinear models [7].

In the multilingual setting, we investigate the effectiveness of sharing different parts of the model. The break-down of the neural machine translation models is illustrated as in Figure 1

- **The embedding layers** project the discrete words into dense vectors. We also consider the output linear layer as an embedding one. These layers are language specific and their parameters cannot be shared across languages.
- **The encoders** encode the representation of the source sentences into a set of vectors S. We can share this component by using one single encoder to encode sentences regardless of the language.
- **The attention layer** reads the encoded source S and learns to focus on important information at every time step which is used for decoding. The attention layer depends on both the source and target languages.
- **The decoder** receives the context information from the attention layer and learns to generate target sentences.

3.1.2. Sharing Embeddings

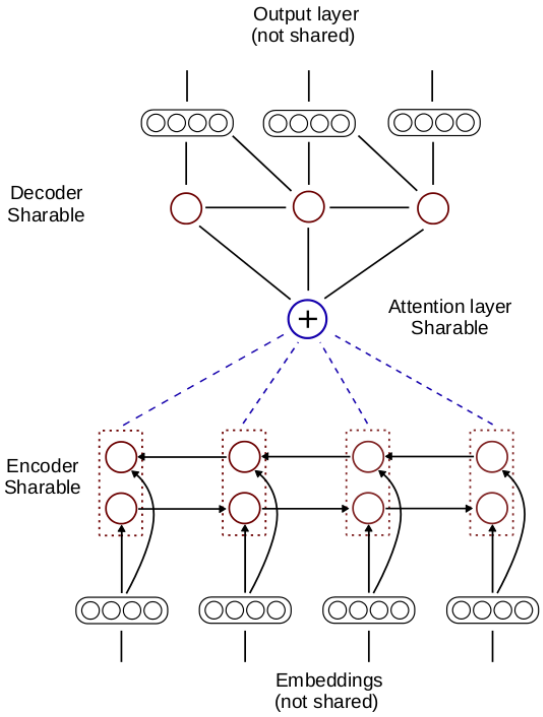
In this multiway, multilingual scenario, we have in total 5 languages on the both source and target sides. We want to ensure that the model has the same view of the embeddings on the source and target side, i.e a German word on the source data has the same embedding values as the same German word on the target sentences. Therefore, we construct one single projection matrix for each language, and use them according to the language of the sentences in the mini-batch.

For the output layer which computes the probabilities of the words, there are two different scenarios: if we use distinct vocabularies for each language, we then end up constructing five different output layers. Because of this architectural choice, each minibatch only contains sentences from one single language pair. In the second scenario, the probability distribution is computed of all words of all languages, then the output layer is not separated as in the first one. The two output layer scenarios are almost equivalent, but the former is much computationally faster than the latter, because the softmax layer required for each mini-batch is considerably smaller.

3.1.3. Sharing Encoder and Decoder

The encoder and decoder are fundamentally built by recurrent neural networks which learn the structural dependency

Figure 1: Neural Machine Translation architecture with shared components



of the words in sentences. For each language at the source and at the target, we assign a separate RNN encoder and a separate decoder. Similar to [4, 2], the specific language encoder weights are only updated when they are used during learning a particular mini-batch. In the sharing scenario, we just need to tie the weights and their gradients to the encoders and decoders.

3.1.4. Sharing Attention Mechanism

The attention layer consists of one feed-forward neural network which connects the hidden layers of the encoders with the hidden layer of the decoders. When being shared, the same network is used across twenty language pairs, while if attention is not shared, each language pair is assigned to one attention layer. Notably, sharing attention has been used in most multilingual setups [4, 3, 5] since the number of attention layers increases quadratically with respect to the number of language pairs, and it is believed that the shared attention layer can benefit better from multilingual sources [4].

4. Speech Translation

4.1. Punctuation Generation

Automatic speech recognition (ASR) systems typically do not generate punctuation marks or reliable casing. Using the raw output of these systems as input to MT causes a performance drop due to mismatched train and test condi-

tions. We used a monolingual NMT system to recase, insert proper punctuation, and add sentence boundaries to ASR output where necessary before translating [9].

To train, we created parallel data where the source sentence is the target sentence lowercased with all punctuation removed. Rare words were replaced with POS tags. The training data was randomly segmented so that segment boundaries and punctuation types were well-distributed throughout the corpus. For the English→German and German→English lecture data, segment boundaries are given, but for TED, they are not. At test time, we used a sliding window of length 10 to observe each word in multiple contexts as described in [9].

We used single-layer biLSTMs for the encoder and decoder, with 256 hidden units for the encoder/decoder/attention layers. Models were trained with Adam. We restarted the algorithm twice and applied early stopping.

4.2. Noised Training

Our speech translation model is applied to noisy and erroneous speech recognition outputs, despite never having been exposed to noisy data during the training process. The result is a harmful mismatch between training and test data that further aggravates the difficulty of having to transform malformed inputs in the first place. Sequence-to-sequence models have been observed to be especially sensitive to corrupted inputs due to erroneous ASR [10]. To improve robustness at test-time, we experiment with inducing a suitable form of noise during the training process. Specifically, we corrupt the source side of the parallel training data by randomly introducing substitution, insertion, and deletion errors. In this way, training data is made more similar to the testing condition, and the model potentially learns to handle noisy inputs at test-time in a more robust fashion.

The noise model is described in detail in [11]. Here, we used the simplified noise model sampling deletions only, at a noise rate of $\tau = 0.01$.

5. Results and Analysis

In this section, we present a summary of our experiments we have carried out for the IWSLT 2017 evaluation[12]. All the reported scores are case-sensitive BLEU scores.

5.1. Machine Translation tracks

5.1.1. Training details

System overview We built a neural machine translation framework which is customized with multiple encoders-decoders-attention for this multilingual task using PyTorch [1]. For the small data task, we use a small network configuration with word embedding and hidden layer size of 512 for all experimented architectures, except for the Share-All one which we found that layer size of 1024 is required to avoid underfitting. For the big data task, all of the models are trained with a larger config, with layer size of 1024. We applied Dropout between the vertical connection of the recurrent networks [13] with probability 0.5. We sampled minibatches containing sentences from only one language-pair so that the model can observe all sentences once every epoch. The parameters are updated using Adam optimizer [14] with the gradients clipped at 5. It is noteworthy that certain models with separated components can suffer from sparse updates since the unused components gradients are treated normally by Adam for the stat computation steps. We observed the training progress with the average *perplexity* on the validation sets, and used the models with the lowest perplexity to translate the test sets.

Adaptation We employed two different strategies of adaptation: in-domain (only applicable for large data task) and language-specific adaptation. Concretely, for in-domain adaptation after our models converge on the training set, we fine-tuned them further on the TED data as proposed in [15]. For language-specific adaptation, after we obtain the best performing model on the validation data, we continue training on each language pair. In the later section, the experimental results indicate that the language-specific adaptation is beneficial.

5.1.2. Main Results

For both tasks, we report the system performance on the test set with the tokenized BLEU (tBLEU) as well as the case-sensitive BLEU (cBLEU) scores. We explore three different architectures, based on our model design described in Section 3.1:

- Share-All : We tie all parameters of the encoders, decoders and attention layers across language pairs
- Share-RNN : The encoders and decoders parameters are shared, but explicit attention layers are separated for each language pair
- Separate-All : Encoders and decoders are language-specific, and the attention layers are separated.

Besides, we also employed the multilingual architecture from [5], here after referred to as 'Language-coded Multilingual'. The most similar architecture to Language-coded Multilingual is Share-All, where all components of the NMT system are shared. Language-coded Multilingual relies on preprocessing steps to share information while keeping the NMT architecture unchanged. In Language-coded Multilingual, however, the output is a big softmax layer, considering all distinct target words in all languages at the same time. Thus, Language-coded Multilingual is quite expensive to train and decode compared to our aforementioned architecture. We reported the result of Language-coded Multi-

[1] http://pytorch.org/

Table 1: Average BLEU scores on the test set for Small task

System	tokenized BLEU	case-sensitive BLEU
Separate-All	24.7	22.6
+ Lang-adapted	25.8	24
Share-RNN	26.0	24.2
+ Lang-adapted	26.3	**24.5**
Share-All	25.2	23.5
+ Lang-adapted	26.2	24.2
Language-coded Multilingual	25.6	23.8
Share-All + Lang-adapted + Average	25.7	23.8
Ensemble	27.4	**25.6**

lingual only on the Small task and without any adaptation scheme.

We also applied some strategies on top of Language-coded Multilingual systems to effectively improve the zero-shot translation. First we built two Language-coded Multilingual-based *Zero* systems, one used 18 language pairs excepts German↔Dutch, the other used 18 language pairs excepts Italian↔Romanian following the architecture suggested by [5]. Then we built other systems employing two strategies: *Target Dictionary Filtering* and *Language as a Word Feature* For greater details of those strategies, please refer to [16].

Small task The translation scores on the test data reflected that sharing the RNN encoders and decoders is clearly effective in multilingual setups. Both the architectures with shared RNNs outperformed their Separate-All counterpart, by 0.9 and 1.6 cBLEU. For the attention mechanism, we found out that sharing the attention reduces translation performance by 0.7 BLEU. Even with the shared recurrent networks, the context vectors from different languages are distinguishable, which is advantageous for the separate attention layers.

Also, as illustrated from table 2, language-specific adaptation helped us to improve the score, which is most clearly seen on the Separate-All model. The gain is also observed on the other two architectures, but not significant. This finding is in-line with [17], which shows that task-specific adaptation is necessary in for multi-task learning with neural encoder-decoders. Our final system to be submitted is the ensemble of three models after adaptation. Notably, the ensemble of Share-All and Share-RNN yields the same performance as the ensemble of all six models, showing that the adapted model dominates the others.

Meanwhile, the Language-coded Multilingual model performed best. Unsurprisingly, the scores from that system are similar to Share-All's. Due to its expensive training and different preprocessing pipeline, however, we did not attempt to employ adaptation and ensemble on that architecture.

The language-specific adaptation method is disadvantageous in that we have to store one model for each direction. Therefore, we tried to take all of the 20 models and average their parameters; interestingly, the averaged model performs better than the pre-adapted one.

Large task Moving over the large data set, we observe the same phenomenon as the small one, in which the Separate architecture fell behind the other two. Interestingly, Shared-All and Shared-RNN produce the same translation performance. One reason why may be that the shared-attention mechanism requires more data to become robust to language-specific mappings.

Even after adaptation (TED in-domain and language specific), the addition of the Europarl corpus only manages to improve the BLEU score by 0.4 for the best system. However, we reckon that further improvement can be achieved by increasing the model size and better parameter search, as was observed in the Small task.

Table 2: Average BLEU scores on the test set for Large task All systems are language-specifically adapted

System	tokenized BLEU	case-sensitive BLEU
Share-All	26.9	25.1
Share-RNN	26.9	25.1
Separate-All	25.1	23.4
Ensemble	27.8	**26.0**

Zero-shot task. We conducted the zero-shot translation for 4 directions asked by IWSLT'17 organizers: German↔Dutch and Italian↔Romanian. The results are shown in Table 3. We can see that *Language as a Word Feature* greatly improves our zero-shot translation systems.

5.2. Spoken Language Translation tracks

Our main translator for this task is the multilingual Share-All model trained with large data (which is also adapted on TED

System	DE→NL		NL→DE		IT→RO		RO→IT	
	dev2010	tst2010	dev2010	tst2010	dev2010	tst2010	dev2010	tst2010
Zero [5]	15.87	19.46	14.03	19.59	11.61	15.44	16.18	17.11
Zero Filtered Dict	15.79	19.48	13.96	19.59	11.52	15.45	16.21	17.20
Zero Lang Feature	**16.65**	**19.68**	**14.50**	**20.67**	**12.70**	**16.22**	**17.26**	**17.79**

Table 3: Effectiveness of proposed strategies on performance of zero-shot translation systems

data as well as language-specific data). However, there is a mismatch between the cleaned text data which is used for MT training and the noisy speech recognition output which can be disfluent, repetitive or lack of punctuations. Therefore, our effort to alleviate this problem is to apply a noisy model on the TED training data and then to adapt the translation models, as described in section 4.2.

The model is tuned with noisy data for ten more epochs (due to sampling, the data is actually slightly modified after each epoch) with learning rate of 0.0001. We conduct experiments on tst2013 sets on two directions: German→English and English→German. The experimental results are shown in Table 4 with case-sensitive BLEU scores. Using the noise models, we can improve the translation scores on both test sets by 0.5 and 0.3 respectively.

5.3. Other findings

In this section, we report the experimental findings that were not considered in the submission systems, including the configurations that we did not afford to finish.

Model capacity Initially we used layer size of 512 for all models for the Small task. With such capacity, the *Separate* was indeed the best model. However, when we scale the layer size to 1024, both of the two shared models improved drastically while the *Separate* model suffered from over-fitting despite of the high Dropout value. We express that, in order to fit the amount of training data that is quadratically larger than a single direction, the model capacity also needs to be scaled accordingly.

Such observation can also potentially explain the lackluster of the models trained on Large data. Such amount of data probably requires a larger/deeper model to utilise, which has been empirically experimented by [3]. However, as the larger model is much slower to train, we decided to keep the same configuration to have a reasonable training time.

Dropout Dropout is also one of the important factor to the model quality. We found out that on the small data, albeit the training data is 20 times larger than a single direction, a larger dropout value of 0.5 helps the model to regularise better than lower values such as 0.2, which is applicable for all architectures.

BPE size In the earlier experiments on the TED data, we tried out different BPE sizes of 40000 and 80000 merging operations, which were done over the concatenated data of all languages. We did not see any improvement of translation and proceeded to use 40000 in the later experiments.

Table 4: BLEU scores on tst2013 for Spoken Language Translation task

System	tst2013 EN-DE	tst2013 DE-EN
baseline	17.9	15.7
noise	18.4	16.0

6. Conclusions

In this paper, we described several innovative techniques that we applied to our multilingual neural machine translation systems, submitted to the IWSLT 2017 Evaluation Campaign. In order to use a single multilingual system instead of many individual systems, we tailored a standard neural translation framework to perform multi-task learning, where each language takes the role of one task. By doing so, we investigated different architectures with different shared components. Our experiments show that ensembling those systems improves the translation performance of the multilingual task further. In addition, a new training technique, the noise model, proved to be beneficial in the SLT task by making the translation system more robust on spoken data.

7. Acknowledgements

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement n° 645452. The research by Thanh-Le Ha was supported by Ministry of Science, Research and the Arts Baden-Württemberg.

8. References

[1] R. Sennrich, B. Haddow, and A. Birch, "Neural Machine Translation of Rare Words with Subword Units," in *Association for Computational Linguistics (ACL 2016)*, Berlin, Germany, August 2016.

[2] M.-T. Luong, Q. V. Le, I. Sutskever, O. Vinyals, and L. Kaiser, "Multi-task sequence to sequence learning," in *International Conference on Learning Representations (ICLR)*, San Juan, Puerto Rico, May 2016.

[3] M. Johnson, M. Schuster, Q. V. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. B. Viégas, M. Wattenberg, G. Corrado, M. Hughes, and J. Dean, "Google's multilingual neural machine translation system: Enabling zero-shot translation," *CoRR*, vol. abs/1611.04558, 2016. [Online]. Available: http://arxiv.org/abs/1611.04558

[4] O. Firat, K. Cho, and Y. Bengio, "Multi-Way, Multilingual Neural Machine Translation with a Shared Attention Mechanism," *CoRR*, vol. abs/1601.01073, 2016. [Online]. Available: http://arxiv.org/abs/1601.01073

[5] T.-L. Ha, J. Niehues, and A. Waibel, "Toward Multilingual Neural Machine Translation with Universal Encoder and Decoder," *Proceedings of the 13th International Workshop on Spoken Language Translation (IWSLT 2016)*, 2016.

[6] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," *CoRR*, vol. abs/1409.0473, 2014. [Online]. Available: http://arxiv.org/abs/1409.0473

[7] M.-T. Luong, I. Sutskever, Q. V. Le, O. Vinyals, and W. Zaremba, "Addressing the rare word problem in neural machine translation," in *Proceedings of ACL-IJNLP 2015*. Beijing, China: Association for Computational Linguistics, July 2015, pp. 11–19. [Online]. Available: http://www.aclweb.org/anthology/P15-1002

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[9] E. Cho, J. Niehues, and A. Waibel, "Nmt-based segmentation and punctuation insertion for real-time spoken language translation," *Proc. Interspeech 2017*, pp. 2645–2649, 2017.

[10] N. Ruiz, M. A. Di Gangi, N. Bertoldi, and M. Federico, "Assessing the Tolerance of Neural Machine Translation Systems Against Speech Recognition Errors," in *Annual Conference of the International Speech Communication Association (InterSpeech)*, Stockholm, Sweden, 2017, pp. 2635–2639.

[11] M. Sperber, J. Niehues, and A. Waibel, "Toward toward robust neural machine translation for noisy input sequences," in *Proceedings of the 14th International Workshop on Spoken Language Translation (IWSLT 2017)*, Tokyo, Japan, 2017.

[12] M. Cettolo, M. Federico, L. Bentivogli, J. Niehues, S. Stüker, K. Sudoh, K. Yoshino, and C. Federmann, "Overview of the IWSLT 2017 Evaluation Campaign," in *Proceedings of the 14th International Workshop on Spoken Language Translation (IWSLT)*, Tokyo, Japan, 2017.

[13] V. Pham, T. Bluche, C. Kermorvant, and J. Louradour, "Dropout improves recurrent neural networks for handwriting recognition," in *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*. IEEE, 2014, pp. 285–290.

[14] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[15] M.-T. Luong and C. D. Manning, "Stanford neural machine translation systems for spoken language domains," in *Proceedings of the International Workshop on Spoken Language Translation*, 2015.

[16] T.-L. Ha, J. Niehues, and A. Waibel, "Effective Strategies in Zero-Shot Neural Machine Translation," *Proceedings of the 14th International Workshop on Spoken Language Translation (IWSLT 2017)*, 2017.

[17] J. Niehues and E. Cho, "Exploiting linguistic resources for neural machine translation using multi-task learning," *CoRR*, vol. abs/1708.00993, 2017. [Online]. Available: http://arxiv.org/abs/1708.00993