Effective Strategies in Zero-Shot Neural Machine Translation

Thanh-Le Ha, Jan Niehues, Alexander Waibel

Institute for Anthropomatics and Robotics
KIT - Karlsruhe Institute of Technology, Germany
firstname.lastname@kit.edu

Abstract

In this paper, we proposed two strategies which can be applied to a multilingual neural machine translation system in order to better tackle zero-shot scenarios despite not having any parallel corpus. The experiments show that they are effective in terms of both performance and computing resources, especially in multilingual translation of unbalanced data in real zero-resourced condition when they alleviate the language bias problem.

1. Introduction

The newly proposed neural machine translation [1] has shown the best performance in recent machine translation campaigns for several language pair. Being applied to multilingual settings, neural machine translation (NMT) systems have been proved to be benefited from additional information embedded in a common semantic space across languages. However, in the extreme cases where no parallel data is available to train such system, often NMT systems suffer a bad training situation and are incapable to perform adequate translation.

In this work, we point out the underlying problem of current multilingual NMT systems when dealing with zero-resource scenarios. Then we propose two simple strategies to reduce adverse impact of the problem. The strategies need little modifications in the standard NMT framework, yet they are still able to achieve better performance on zero-shot translation tasks with much less training time.

1.1. Neural Machine Translation

In this section, we briefly describe the framework of Neural Machine Translation as a sequence-to-sequence modeling problem following the proposed method of [1].

Given a source sentence $\boldsymbol{x} = (x_1, .., x_i, .., x_I)$ and the corresponding target sentence $\boldsymbol{y} = (y_1, .., y_j, .., y_J)$, the NMT aims to directly model the translation probability of the target sequence:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \prod_{j=1}^{I} P(y_j|y_{<j}, \boldsymbol{x}; \boldsymbol{\theta})$$

[1] proposed an *encoder-attention-decoder* framework to calculate this probability.

A bidirectional recurrent *encoder* reads a word x_i from the source sentence and produces a representation of the sentence in a fixed-length vector $\boldsymbol{h}_i$ concatenated from those of the forward and backward directions:

$$\boldsymbol{h}_i = [\overrightarrow{\boldsymbol{h}}_i, \overleftarrow{\boldsymbol{h}}_i]$$
$$\overrightarrow{\boldsymbol{h}}_i = d(\overrightarrow{\boldsymbol{h}}_{i-1}, \boldsymbol{E}_s \bullet \boldsymbol{x}_i) \quad (1)$$
$$\overleftarrow{\boldsymbol{h}}_i = d(\overleftarrow{\boldsymbol{h}}_{i+1}, \boldsymbol{E}_s \bullet \boldsymbol{x}_i) \quad (2)$$

where $\boldsymbol{E}_s$ is the source word embedding matrix to be shared across the source words $x_i \in V_x$, d is the recurrent unit computing the current hidden state of the encoder based on the previous hidden state. $\boldsymbol{h}_i$ is then called an *annotation vector* which encodes the source sentence up to the time i from both forward and backward directions.

Then an *attention mechanism* is set up in order to choose which annotation vectors should contribute to the predicting decision of the next target word. Normally, a relevance score $rel(\boldsymbol{z}_{j-1}, \boldsymbol{h}_i)$ between the previous target word and the annotation vectors is used to calculate the context vector $\boldsymbol{c}_i$:

$$\alpha_{ij} = \frac{\exp(rel(\boldsymbol{z}_{j-1}, \boldsymbol{h}_i))}{\sum_{i'} \exp(rel(\boldsymbol{z}_{j-1}, \boldsymbol{h}_{i'}))}, \ \boldsymbol{c}_j = \sum_i \alpha_{ij} \boldsymbol{h}_i$$

In the other end, a *decoder* recursively generates one target word y_j at a time:

$$P(y_j|y_{<j}, \boldsymbol{x}; \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{z}_j)}{\sum_{k=1}^{|V_y|} \exp(\boldsymbol{z}_k)}$$

Where:

$$\boldsymbol{z}_j = g(\boldsymbol{z}_{j-1}, \boldsymbol{t}_{j-1}, \boldsymbol{c}_j)$$
$$\boldsymbol{t}_{j-1} = \boldsymbol{E}_t \bullet \boldsymbol{y}_{j-1} \quad (3)$$

The mechanism in the decoder is similar to its counterpart in the encoder, excepts that beside the previous hidden state $\boldsymbol{z}_{j-1}$ and target embedding $\boldsymbol{t}_{j-1}$, it also takes the context vector $\boldsymbol{c}_j$ from the attention layer as inputs to calculate the current hidden state $\boldsymbol{z}_j$. The predicted word y_j at time j then can be sampled from a softmax distribution of the hidden state. Basically, a beam search is utilized to generate the output sequence - the translated sentence in this case.

Original corpus		
Source Sentence 1	De	versetzen Sie sich mal in meine Lage !
Target Sentence 1	En	put yourselves in my position .
Source Sentence 2	En	I flew on Air Force Two for eight years .
Target Sentence 2	Nl	ik heb acht jaar lang met de Air Force Two gevlogen .
Preprocessed by [2]		
Source Sentence 1	De	**<en> <en>** *de_*versetzen *de_*Sie *de_*sich *de_*mal *de_*in *de_*meine *de_*Lage *de_*! **<en> <en>**
Target Sentence 1	En	*en__ en_*put *en_*yourselves *en_*in *en_*my *en_*position *en_*.
Source Sentence 2	En	**<nl> <nl>** *en_*I *en_*flew *en_*on *en_*Air *en_*Force *en_*Two *en_*for *en_*eight *en_*years *en_*. **<nl> <nl>**
Target Sentence 2	Nl	*nl__ nl_*ik *nl_*heb *nl_*acht *nl_*jaar *nl_*lang *nl_*met *nl_*de *nl_*Air *nl_*Force *nl_*Two *nl_*gevlogen *nl_*.
Preprocessed by [3]		
Source Sentence 1	De	**2en** versetzen Sie sich mal in meine Lage !
Target Sentence 1	En	put yourselves in my position .
Source Sentence 2	En	**2nl** I flew on Air Force Two for eight years .
Target Sentence 2	Nl	ik heb acht jaar lang met de Air Force Two gevlogen .

Table 1: Examples of preprocessing steps conducted by [2] and [3].

1.2. Multilingual NMT

State-of-the-art NMT systems have demonstrated that machine translation in many languages can achieve high quality results with large-scale data and sufficient computational power[4, 5]. On the other hand, how to prepare such enormous corpora for low-resourced languages and specific domains has remained a big problem. Especially in zero-resourced condition where we do not possess any bilingual corpus, building a data-driven translation system requires special techniques that can enable some sort of transfer learning. A simple but effective approach called pivot-based machine translation has been developed. The idea of the pivot-based approach is to indirectly learn the translation of the source and target languages through a bridge language. However, this pivot approach is not ideal since it is necessary to build two different translation systems for each language pair in order to perform the bridge translation, hence possibly produces more ambiguities cross languages as well as error-prone to the individual systems.

Recent work has started exploring potential solutions to perform machine translation for multiple language pairs using a single NMT system. One of the most notable differences of NMT compared to the conventional statistical approach is that the source words can be represented in a continuous space in which the semantic regularities are induced automatically. Being applied to multilingual settings, NMT systems have been proved to be benefited from additional information embedded in a common semantic space across languages, thus, by some means they are able to conduct some level of transfer learning.

In this section, we review the related work on constructing a multilingual NMT system involved in translating from several source languages to several target languages. Then we consider a potential application of such a multilingual system on zero-shot scenarios to demonstrate the capability of those systems in extreme low-resourced conditions.

We can essentially divided the work into two directions in applying the current NMT framework for multilingual scenarios. The first direction follows the idea that multilingual training of an NMT system can be seen as a special form of multi-task learning where each encoder is responsible to learn an individual modality's representation and each decoder's mission is to predict labels of a particular task. In such a multilingual system, each task or modality corresponds to a language. In [6], the authors utilizes a multiple encoder-decoder architecture to do multi-task learning, including many-to-many translation, parsing and image captioning. [7] proposed another approach which enable attention-based NMT to multilingual translation. Similar to [6], they use one encoder per source language and one decoder per target language for *many-to-many* translation tasks. Instead of a quadratic number of independent attention layers, however, their NMT system contains only a single, huge attention layer. In order to achieve this, the attention layer need to be provided some sort of aggregation layer between it and the encoders as well as the decoders. It is required to change their architecture to accommodate such a complicated shared attention mechanism.

The work along the second direction also considers multilingual translation as multi-task learning, although the tasks should be the same (i.e. translation) with the same modality (i.e. textual data). The only difference here is whether we decide which components are shared across languages or we let the architecture learns to share what. In [8], the authors developed a general framework to analyze which components should be shared in order to achieve the best multilingual translation system. Other works chose to share every components by grouping all language vocabularies into a large vocabulary, then use a single encoder-decoder NMT system

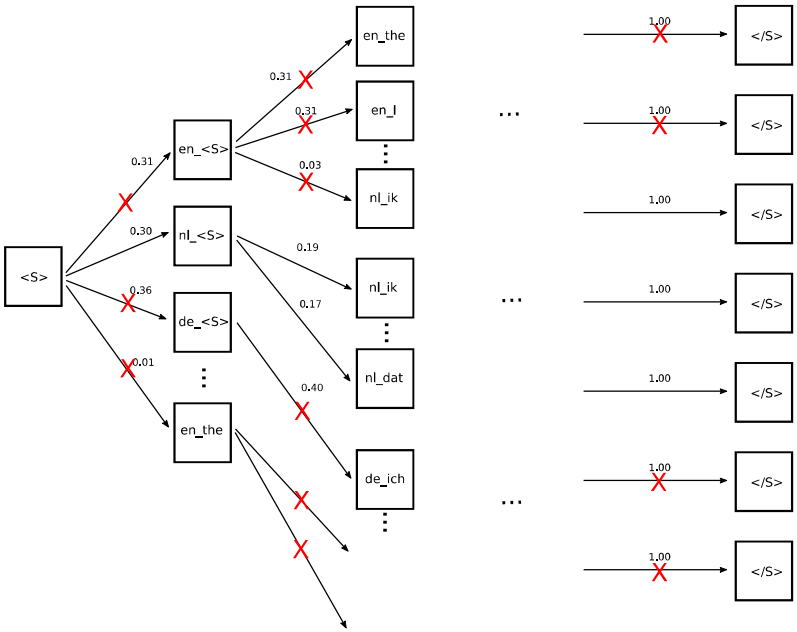

Figure 1: Effect of target dictionary filtering on the decoding process using beam search.

to perform many-to-many translation as each word is viewed as a distinct entry in the large vocabulary regardless of its language. By implementing such mechanism in the preprocessing step, those approaches require little or no modification in the standard NMT architecture. In our previous work[2], we performed a two-step preprocessing:

1. *Language Coding*: Add the language codes to every word in source and target sentences.

2. **Target Forcing**: Add a special token in the beginning of every source sentence indicating the language they want the system to translate the source sentence to.[1]

Concurrently, [3] proposed a similar but simpler approach: they carried out only the second step as in the work of [2]. They expected that there would be only a few cases where two words in difference languages (with different meanings) having the same surface form. Thus, they did not conduct the first step. An interesting side-effect of not doing language-code adding, as [3] suggested, is that their system could accomplish code-switching multilingual translation, i.e. it could translate a sentence containing words in different languages. The main drawback of these approaches is that the sizes of the vocabularies and corpus grow proportionally to the number of languages involved. Hence, a huge amount of time and memory are necessary to train such a multilingual system. Table 1 gives us a simple example illustrating those preprocessing steps.

[1] In fact, we add the target language token both to the beginning and to the end of every source sentence, each place two times, to make the forcing effect stronger. Furthermore, every target sentence starts with a pseudo word playing the role of a start token in a specific target language. This pseudo word is later removed along with sub-word tags in post-processing steps.

2. Multilingual-based Zero-Shot Translation

In this section, we follow the second direction of [2] and [3], hereby called *mix-language* approaches. First we built some baselines inspired of their approaches and participated in the new challenge of zero-shot translation at IWSLT 2017. Then we proposed two strategies, *filtered dictionary* and *language as a word feature*, in attempts to tackle the drawbacks of their approaches. The results in section 3.3 show that our strategies are highly effective in terms of both performance and training resources.

2.1. Target Dictionary Filtering

In [2], the authors discussed about observations of the language bias problem in our multilingual system: If the very first word is wrongly translated into wrong language, the following picked words are more probable in that wrong language again. The problem is more severe when the mixed target vocabulary is unbalanced, due to the language unbalance of the training corpora (whereas the zero-shot is a typical example). We reported a number of 9.7% of the sentences wrongly translated in our basic zero-shot German→French system.

One solution for this problem is to enhance the balance of the corpus by adding *target→target* corpora into the mul-

tilingual system as suggested in [2]. The beam search still need to consider, however, other candidates belonging to the target vocabulary that should not be considered. In this work, we propose a simple yet effective technique to eliminate this bad effect. In the translation process to a specific language, we filter out all the entries in the languages other than that desired language from the target vocabulary. It would significantly reduce the translation time in huge multilingual systems or big texts to be translated due to the fact that many search paths containing the unwanted candidates are removed. More importantly, it assures the translated words and sentences are in the correct language. The effect of this strategy in the decoding process is illustrated in Figure 1.

2.2. Language as a Word Feature

As briefly mentioned in Section 1.2, the main disadvantage of the *mix-language approaches* is the efficiency of training process. Usually in those systems, source and target vocabularies have a huge number of entries, in proportion to the number of languages whose corpora are mixed. It leads to immerse numbers of parameters laying between the embedding and hidden states of the encoder and the decoder. More problematic is the size of the output softmax - where most calculations take place.

There exist works on integrating linguistic information into NMT systems in order to help predict the output words[9, 10, 11]. In those works, the information of a word (e.g. its lemma or its part-of-speech tag) are integrated as a word features. It is conducted simply by learning the feature embeddings instead of the word embeddings. In other words, their system considers a word as a special feature together with other features of itself.

More specially, in the formula 1, 2 and 3, the embedding matrices are the concatenation of all features' embeddings:

$$\boldsymbol{E} \bullet \boldsymbol{x} = \Big\Big[, \Big]_{f \in F} (\boldsymbol{E}^f \bullet \boldsymbol{x}^f)$$

Where $\Big[, \Big]$ is the vector concatenation operation, concatenating the embeddings of individual feature f in a finite, arbitrary set F of word features. The target features of each target word would be jointly predicted along the word. Figure 2 denotes this modified architecture.

Inspired by their work, we attempt to encode the language information directly in the architecture instead of performing language token attachment in the preprocessing step. Being applied in our model, instead of the linguistic information at the word level, our source word features are the language of the considering word and the correct language the target sentence The only target feature is the language of the produced word by the system. For example, when we would like to translate from the sentence "*put yourselves in my position*" into German, the features of each source word would be the word itself, e.g. "*yourselves*", and two additional features "*en*" and "*de*". Similarly, the features of the target words are

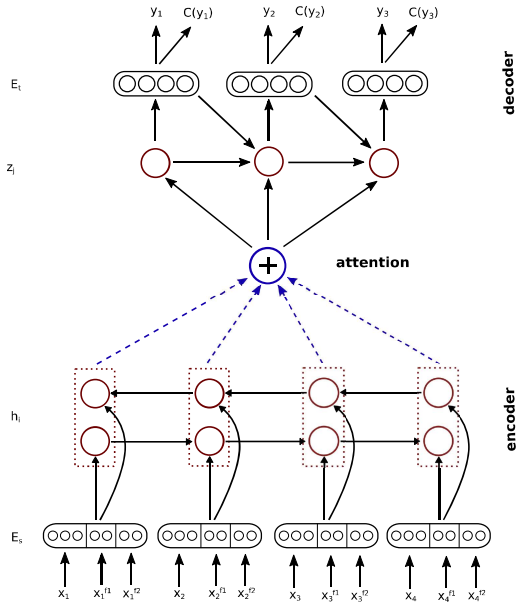

Figure 2: The NMT architecture which allows the integration of linguistic information as word features.

the word and "*de*". This scheme of using language information looks alike to [2], but the difference is the way the language information are integrated into the NMT framework. In [2], those information are implicitly injected into the system. In this work, they are explicitly provided along with the corresponding words. Furthermore, when being used together in the embedding layers, they can share useful information and constraints which would be more helpful in choosing both correct words and language to be translated to. During decoding, the beam search is only conducted on the target words space and not on the target features. When the search is complete, the corresponding features are selected along the search path. In our case, we do not need the output of the target language features excepts for the evaluation of language identification purpose.

3. Evaluation

In this section, we describe a thorough evaluation of the related methods in comparisons with the direct approach as well as the pivot-based approach.

3.1. Experimental Settings

We participated to this year's IWSLT zero-shot tasks for German→Dutch and German→Romanian[12]. The pivot language used in our experiments is English and the parallel corpora are German-English and English-Dutch or German-English and English-Romanian. The data are extracted from

	System	Zero-shot?	German→Dutch		German→Romanian	
			dev2010	tst2010	dev2010	tst2010
(1)	Direct	No	17.83	20.49	12.41	15.14
(2)	Pivot (via English)	Yes	16.11	19.12	12.88	15.04
(3)	Zero 2L [3]	Yes	4.79	5.75	1.55	2.05
(4)	Zero 4L [3]	Yes	6.31	7.93	3.15	3.73
(5)	Zero 6L [2]	Yes	11.58	14.95	8.61	10.83
(6)	Back-Trans [2]	No	17.33	20.36	12.92	15.62

Table 2: Results of the popular *mix-language* methods applied to German→Dutch and German→Romanian zero-shot tasks.

WIT3's[2] TED corpus[13]. The validation and test sets are `dev2010` and `tst2010` which are provided by the IWSLT17 organizers.

We use the Lua version of `OpenNMT`[3][14] framework to conduct all experiments in this paper. Subword segmentation is performed using Byte-Pair Encoding [15] with 40000 merging operations. All sentence pairs in training and validation data which exceeds 50-word length are removed and the rest are shuffled inside each of every minibatch. We use 1024-cell LSTM layers[16] and 1024-dimensional embeddings with dropout of 0.3 at every recurrent layers. The systems are trained using Adam[17]. In decoding process, we use a beam search with the size of 15.

3.2. Baseline Systems

Let us consider the scenario that we would like to translate from a *source* language to a *target* language via a *pivot* language. In order to evaluate the effectiveness of our proposed strategies, we reimplemented the following baseline systems:

- *Direct*: A system which does not exist in the real world is trained using the parallel corpus. It is only for comparison purpose.

- *Pivot*: A system which uses English as the pivot language. The output of the first *source→pivot* translation system was pipelined into the second system trained to translate from *pivot* to *target*.

- *Zero 2L*: To build this system, we followed the idea of [3]: we added a target token to every *source* sentences in the parallel corpus of *source→pivot*, added another target token to every *pivot* sentences in the parallel corpus of *pivot→target*, merged those two parallel corpora into a big corpus and used our standard NMT architecture mentioned in previous section to train and decode. The only differences are the actual data and a simpler NMT architecture we used to train the system.

- *Zero 4L*: Same as *Zero 2L* but in addition applying to two other directions *pivot→source* and *target→pivot*. The result is a parallel corpus two times larger than the corpus in *Zero 2L*.

- *Zero 6L*: This is an extended version of our previous work[2]. There are two main differences compared *Zero 2L* and *Zero 4L*: we conducted both Language Coding and Target Forcing preprocessing steps, the data used to trained are actually six parallel corpora: *source↔pivot*, *pivot→pivot*, *pivot↔target*, *target→target*. Finallly we merged them at the end to form a big parallel corpus.

- *Back-Trans*: This is not a real zero-shot system where we back-translated the English part of the *pivot-target* parallel corpus using a *target-pivot* NMT system. At the end we have a *source-target* parallel corpus with back-translation quality. After we obtained that direct corpus, we apply the same steps as in the *Zero 6L* setting to all corpora we have (8 parallel corpora in total).

3.3. Results

First we applied the baseline systems with respect to the IWSLT17 zero-shot tasks. From Table 2 we can see that in general, translating from German→Romanian is more difficult than German→Dutch, which is reasonable when German and Dutch are considered to be similar. The direct approach which uses a parallel German-*target* corpus and the pivot approach have similar performance in term of BLEU score[18]. Interestingly, the *Back-Trans* performed better that the direct approach on German→Romanian. We spectaculate that back translation might pose some translation noise which makes the translation from German→Romanian more robust.

Compared to the *Zero 6L* model (5), two other Google-inspired models *Zero 2L* (3) and *Zero 4L* (4) from [3] achieved quite low scores. This explains the language-bias problem when these models used less and unbalanced corpora than the *Zero 6L* system. However, the real zero-shot systems (2, 3, 4, 5), excepts the pivot one (2), performed worse than those using direct parallel corpora (1) and (7), since the zero-shot systems have not been shown the direct data, hence, having little or no guide to learn the translation. Among those real zero-shot non-pivot systems, the *Zero 6L*

[2] `https://wit3.fbk.eu/`
[3] `http://opennmt.net/`

	System	Zero-shot?	German→Dutch		German→Romanian	
			dev2010	tst2010	dev2010	tst2010
(1)	Zero 2L [3]	Yes	4.79	5.75	1.55	2.05
(2)	Zero 4L [3]	Yes	6.31	7.93	3.15	3.73
(3)	Zero 6L [2]	Yes	11.58	14.95	8.61	10.83
(3a)	Zero 6L Filtered Dict	Yes	12.50	16.02	9.10	11.00
(3b)	Zero 6L Lang Feature	Yes	13.95	17.15	9.88	11.37
(4)	Back-Trans [2]	No	17.33	20.36	12.92	15.62
(4a)	Back-Trans Filtered Dict	No	17.13	20.22	13.10	15.67
(4b)	Back-Trans Lang Feature	No	17.48	20.24	13.43	15.70

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

system got the best performance due to the amount and the balance of the data used to train. Thus, from hereinafter we consider the *Zero 6L* as the baseline to analyze the effectiveness of our proposed strategies.

When we applied the proposed strategies, it is interesting to see their effects on different types of systems. Since *Zero 2L* and *Zero 4L* do not have the language identity for words, we cannot directly apply our strategies on those systems. In contrast, it is straight-forward to adapt *Target Dictionary Filtering* and *Language as a Word Feature* on the systems described in [2].

Table 3 shows the performance of our strategies compared to [2] and [3] methods. When we applied the strategies on top of *Back-Trans* system, it seems that the data it used to train is sufficient to avoid the language bias problem. Thus, our strategies did not have a significant effect of performance on this system (4a vs. 4 and 4b vs. 4). But on the real zero-shot configuration (3), both strategies helped to improve the systems by notable margins. On `tst2010`, *Target Dictionary Filtering* (3a) brought an improvement of 1.07 on German→Dutch. On the same test set, *Language as a Word Feature* achieved the gains of 2.20 BLEU scores compared to *Zero 6L* (3b vs. 3). On German→Romanian zero-shot task, the improvements of our strategies were not as great as on German→Dutch, but they still helped, especially on `dev2010`.

Table 5 shows two examples where *Target Dictionary Filtering* clearly improves the quality and readability of the translation over the *Zero 6L* when applied.

Considering the effectiveness of our strategy *Language as a Word Feature* on computation perspective, which is shown in Table 4, we observed very positive results. We compared the *Zero 6L* configuration and our *Language as a Word Feature* system in term of training times, size of source&target vocabularies[4] and the total number of model parameters on both zero-shot translation tasks. The models were usually trained on the same GPU (Nvidia Titan Xp) for 8 epochs so they are fairly compared (seeing the same dataset the same number of times). Each type of models has the same configuration between two zero-shot tasks, excepts the parts related to vocabularies[5].

By encoding the language information into word features, the number of vocabulary entries reduces to almost half of the original method. Thus, it leads to the similar reduction in term of the parameter number. This reduction allows us to use bigger minibatches as well as perform faster updates, resulting in substantially decreased training time (from 7.3 hours to 1.5 hours for each epoch in case of German→Dutch and from 6.0 hours to 1.3 hours for each epoch in case of German→Romanian). The strategy requires minimum modifications in the standard NMT framework, yet it still achieved better performance with much less training time.

German→Dutch			
System	#parameters (millions)	Vocab Size (thousands)	Training Time (hours/epoch)
Zero 6L	243	68	7.3
Lang Feature	130	28	1.5
German→Romanian			
System	#parameters (millions)	Vocab Size (thousands)	Training Time (hours/epoch)
Zero 6L	247	69	6.0
Lang Feature	122	31	1.3

Table 4: Effects of the strategy *Language as a Word Feature* on model size and training time.

4. Conclusion and Future Work

In this paper, we present our experiments toward zero-shot translation tasks using a multilingual Neural Machine Translation framework. We proposed two strategies which substantially improved the multilingual systems in terms of both performance and training resources.

On the future work, we would like to look closer to the outputs of the systems in order to analyze better the effects of our strategies. We also have the plan to expand our strategies on full multilingual systems, for more languages and different data conditions.

[4] In all cases, these sizes are similar numbers.

[5] While the total number of parameters on German→Romanian is bigger than that of German→Dutch, the training time of German→Romanian systems is less due to the fact that its training corpus is smaller.

German→Dutch example	
Zero 6L	Een collega van mij had **toegang** tot **investeringsgegevens** van Fox guard
English meaning	A colleague of mine had **access** to **investment data** of Fox guard
Filtered Dict	Een collega van mij had **Zugang** tot **investment** van de autoriteiten van Fox guard
English meaning	A colleague of mine had **Zugang** to **investment** from the authorities of Fox guard
Reference	Een collega van me kreeg **toegang** tot **investeringsgegevens** van Vanguard
English meaning	A colleague of mine received **access** to **investment data** from Vanguard
German→Romanian example	
Zero 6L	Pentru că s-ar aștepta să apelăm la medic în **nächsten** dimineață .
English meaning	Because he would expect to call a doctor in **nächsten** morning .
Filtered Dict	Pentru că s-ar aștepta să-l chemăm pe doctori în **următorul** dimineață .
English meaning	Because he would expect us to call the doctors the **next** morning .
Reference	Răspunsul e că cei care fac asta se așteaptă ca noi să ne sunăm doctorii în dimineața **următoare** .
English meaning	The answer is that people who do this expect us to call our doctors the **following** morning .

Table 5: Examples of the sentences with the words in wrong languages produced by *Zero* systems and the corrected version produced by the same systems having the target dictionary filtered in decoding phase. Target Dictionary Filtering is not only helpful in producing readable and fluent outputs but also clearly affects to the choices of next words.

5. 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.

6. References

[1] 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

[2] T.-L. Ha, J. Niehues, and A. Waibel, "Toward Multilingual Neural Machine Translation with Universal Encoder and Decoder," *CoRR*, vol. abs/1611.04798, 2016. [Online]. Available: http://arxiv.org/abs/1611.04798

[3] M. Johnson, M. Schuster, Q. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. Viégas, M. Wattenberg, G. Corrado, M. Hughes, and J. Dean, "Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 339–351, 2017. [Online]. Available: https://www.transacl.org/ojs/index.php/tacl/article/view/1081

[4] O. Bojar, R. Chatterjee, C. Federmann, B. Haddow, P. Koehn, J. Leveling, C. Monz, P. Pecina, M. Post, H. Saint-Amand, *et al.*, "Findings of the 2016 Conference on Machine Translation (WMT16)," in *Proceedings of the First Conference on Machine Translation (WMT16)*. Berlin, Germany: Association for Computational Linguistics, 2016, pp. 12–58.

[5] M. Cettolo, J. Niehues, S. Stüker, L. Bentivogli, R. Cattoni, and M. Federico, "The IWSLT 2016 Evaluation Campaign," in *Proceedings of the 13th International Workshop on Spoken Language Translation (IWSLT 2016)*, Seattle, WA, USA, 2016.

[6] 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.

[7] 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

[8] N.-Q. Pham, M. Sperber, E. Salesky, T.-L. Ha, J. Niehues, and A. Waibel, "KIT's Multilingual Neural Machine Translation systems for IWSLT 2017," in *Proceedings of the 14th International Workshop on Spoken Language Translation (IWSLT 2017)*, Tokyo, Japan, 2017.

[9] R. Sennrich and B. Haddow, "Linguistic input features improve neural machine translation," *CoRR*, vol. abs/1606.02892, 2016. [Online]. Available: http://arxiv.org/abs/1606.02892

[10] C. D. V. Hoang, R. Haffari, and T. Cohn, "Improving neural translation models with linguistic factors," in *Proceedings of the Australasian Language Technology Association Workshop 2016*, 2016, pp. 7–14.

[11] J. Niehues, T.-L. Ha, E. Cho, and A. Waibel, "Using factored word representation in neural network language models." in *Proceedings of the First Conference*

on Machine Translation (WMT16). Berlin, Germany: Association for Computational Linguistics, 2016.

[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] M. Cettolo, C. Girardi, and M. Federico, "Wit^3: Web inventory of transcribed and translated talks," in *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, Trento, Italy, May 2012, pp. 261–268.

[14] G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. M. Rush, "OpenNMT: Open-Source Toolkit for Neural Machine Translation," *ArXiv e-prints*, 2017.

[15] 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.

[16] 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

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

[18] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL 2002)*. Association for Computational Linguistics, 2002, pp. 311–318.