Factored Neural Machine Translation Architectures

Mercedes Garcia-Martinez, Loïc Barrault, Fethi Bougares

Factored Neural Machine Translation Architectures

Mercedes García-Martínez, Loïc Barrault, Fethi Bougares

LIUM, University of Le Mans, France

firstname.lastname@univ-lemans.fr

Abstract

In this paper we investigate the potential of the neural machine translation (NMT) when taking into consideration the linguistic aspect of target language. From this standpoint, the NMT approach with attention mechanism [1] is extended in order to produce several linguistically derived outputs. We train our model to simultaneously output the lemma and its corresponding factors (*e.g.* part-of-speech, gender, number). The word level translation is built with a mapping function using *a priori* linguistic information. Compared to the standard NMT system, factored architecture increases significantly the vocabulary coverage while decreasing the number of unknown words. With its richer architecture, the Factored NMT approach allows us to implement several training setup that will be discussed in detail along this paper. On the IWSLT'15 English-to-French task, FNMT model outperforms NMT model in terms of BLEU score. A qualitative analysis of the output on a set of test sentences shows the effectiveness of the FNMT model.

1. Introduction

Neural Machine Translation (NMT) approach has been further developed in the last years [1, 2]. In contrast to the traditional phrased-based statistical machine translation [3] that represents and translates the input sentence with a set of phrases, NMT uses the sequence to sequence learning architecture and the whole input sentence is considered as one unit for translation [4]. Recently, NMT is gaining more and more interest and showing better accuracy than phrase-based system translating several language pairs. In spite of these recent improvements, the NMT systems still have some restrictions and difficulties to translate. One of them is the high computational of the softmax function which requires to normalize all the output vocabulary size.

In order to solve this issue, a standard technique is to define a *short-list* containing only the most frequent words. This has the disadvantage of increasing the number of unknown words.

In [5], authors propose to carefully organise the batches so that only a subset K of the target vocabulary is possibly generated at training time. This allows the system to train a model with much larger target vocabulary without substantially increasing the computational complexity. Another alternative is proposed by [6] where a structured output layer (SOUL) is defined to handle the words not appearing in the shortlist. This allows the system to always apply the softmax normalization on a layer with reduced size.

Recently, some works have used subword units level instead of word-level for translation. In [7], the rare and unknown words are encoded as subword units with the Byte Pair Encoding (BPE) algorithm. The authors show that this method can generate words which are unseen at training time. Another lower level for translation is the character-level NMT, which has been presented in several works [8, 9, 10] and showed promising results.

Factored Neural Machine Translation (FNMT) approach handles the output vocabulary size problem using factors as a translation unit. The main motivation behind this factored representation is related to the human way to learn how to construct correct sentences. In this work, the factors are referring to the linguistic annotation at word level like the Part of Speech (POS) tags. Some works have used factors as additional information for language modeling [11] and also applied for neural networks language models [12, 13, 14]. Recently, factors have been used as additional linguistic input features to improve a word-level NMT system [15] as well.

This approach differs from previous works in the sense that it uses only the linguistic decomposition of the words (no surface form word level) and it is applied to the output language. Figure 1 presents the general architecture of our FNMT system where two different outputs are generated: (1) the lemma of the word ; (2) its factors. Indeed, each word is represented by its lemma and its linguistic factors (POS tag, tense, gender, number and person). By these means, the target vocabulary size is reduced because all the derived forms of the verbs, nouns, adjectives, etc are not kept. Furthermore, new words that are not in the vocabulary using all the derived forms of the lemmas are produced.

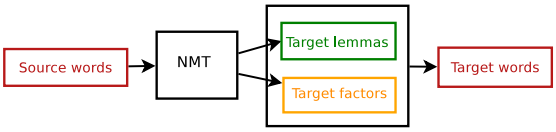

Figure 1: Pipeline of NMT system with Factored output

Multiple output neural networks were previously pro-

posed [16] using scheduled decoders with multiple source and target languages. In contrast to this, the FNMT system simultaneously produces several outputs. Given both outputs (lemma and factors) and linguistic resources, the final surface form is easily generated.

The rest of the paper is structured as follows: Section 2, about background work, describes the based NMT system and FNMT system. In Section 3, we describe the different experiments to go further with the FNMT architecture and to compare its results with the word-level NMT and other state of the art systems as BPE and multilingual NMT. Section 4 presents a deep analysis of the translation output of the FNMT systems. Finally, Section 5 includes the conclusions and future work.

2. Background systems

2.1. Attention-based Neural Machine Translation

The encoder-decoder architecture, used for NMT, consists of two recurrent neural networks (RNN), one for the encoder and the other for the decoder. The encoder maps a source sequence into a sequence of continuous space vectors and the decoder maps this representation back to a target sequence. Our trained neural translation models are based on an encoder-decoder deep neural network, equipped with an attention mechanism [1], as described in Figure 2.

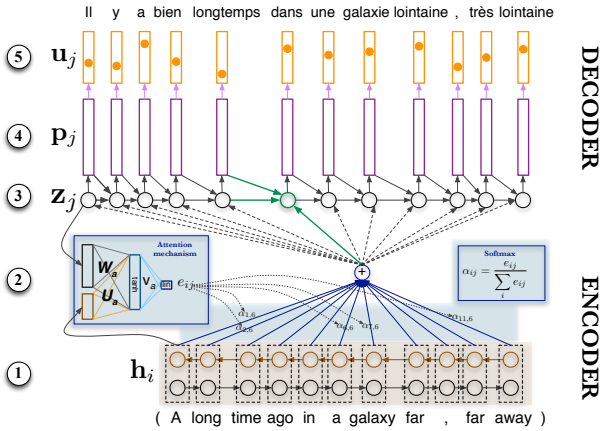

Figure 2: Attention-based NMT system.

This architecture consists of a bidirectional RNN encoder (as seen in stage 1 of Figure 2). An input sentence is encoded in a sequence of annotations (one for each input word), corresponding to the concatenation of the outputs of a forward and a backward RNN. Each annotation represents the full sentence with a strong focus on the current word. The decoder is composed of a conditional RNN as provided for the DL4MT winter school[1] (see stage 3 of Figure 2), equipped with an attention mechanism (stage 2). The attention mechanism aims at providing weights for each annotation in order to generate a context vector (by performing a weighted sum over the annotations). The attention mechanism uses the hidden state at timestep j of the decoder RNN along with the annotation h_i to generate a coefficient e_{ij}. A softmax operation is performed over those coefficients to generate the annotation weights α_{ij}. As described in [1], the annotation weights can be used to align the input words to the output words. The RNN takes as input the context vector, the embedding of the previous output word (stage 4 of Figure 2), and of course, its hidden state. Finally, on stage 5 of the Figure 2, the output probabilities of the target vocabulary are computed. The word with the highest probability is selected to be the translation output at each timestep. The encoder and the decoder are trained jointly to maximize the conditional probability of the correct translation.

2.2. Factored Neural Machine Translation

The Factored neural machine translation is an extension of the standard NMT architecture which allows us generating several output symbols simultaneously as presented in Figure 3. The encoder and attention mechanism of the Figure 2 remain without modifications. However, the decoder has been modified to get multiple outputs[2].

For simplicity reasons, only two symbols are generated: the lemma and the concatenation of the different factors that we consider. For example, from the French word *devient*, we obtain the lemma *devenir* and the factors *VP3#S*, meaning that it is a **V**erb, in **P**resent, **3**rd person, irrelevant gender (**#**) and **S**ingular. The morphological and grammatical analysis is performed with the MACAON toolkit [17]. MACAON POS-tagger outputs the lemma and factors for each word taking into account its context. For the very few cases when MACAON proposes multiple factors, the first proposition is taken.

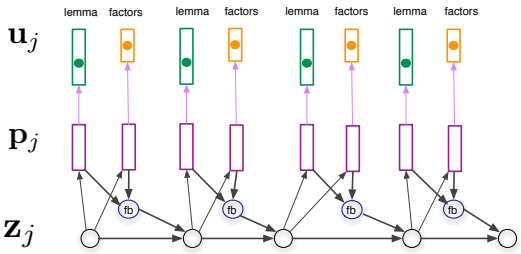

Figure 3: Detailed view of the decoder of the FNMT system.

The decoder of the FNMT architecture presented in Figure 3 may lead to sequences with different length since lemmas and factors are generated in a synchronous stream, but in separate outputs. Indeed, each sequence of symbols ends when the end-of-sequence (<eos>) symbol is generated with this architecture, and nothing prevents the lemma generator to output the <eos> symbol before or after the factors

[1] https://github.com/nyu-dl/dl4mt-tutorial

[2] The source code is available at *https://github.com/merc85garcia/fnmt*

generator. To avoid this scenario, the length of the factors sequence is constricted to be equal to the length of the lemma sequence. This implies that to ignore the <eos> symbol for factors (to avoid shorter factors sequence) and stop the generation of factors when the lemma sequence has ended (to avoid longer factors sequence). This is motivated by the fact that the lemmas are closer to the final objective (a sequence of **words**) and that they are the symbols carrying most of the meaning.

In the NMT approach, the previous word is given as feedback for generating the next word. For the FNMT approach, multiple outputs are available which can be given as feedback. Consequently, several options are possible and will be explored in Section 3.3.

Once we obtain the factored outputs from the neural network, the post-process to fall back to the word representation is performed. This operation is also performed with the MACAON tool, which given a lemma and some factors, provides the word candidate. In the cases (*e.g.* name entities) that the word corresponding to the lemma and factors is not found, the system outputs the lemma itself.

3. Experiments

In order to study the behavior of the Factored model, we will explore different architectures and show their outcome. We study the different options of feedback since we have two outputs and, therefore, several combinations of these two values are possible for feedback. Moreover, we implemented a dependency in factors output in order to improve its performance. We compare them with the word-based NMT, subwords-based NMT and multiway, multilingual NMT systems.

3.1. Data processing and selection

We evaluate our experiments on the English to French Spoken Language Translation task from IWSLT 2015 evaluation campaign[3]. A data selection method [18] consisting on scoring the sentences according to a in-domain language model has been applied. We have used as available parallel corpora (news-commentary, united-nations, europarl, wikipedia, and two crawled corpora) and Technology Entertainment Design (TED[4]) corpus as in-domain corpus. The data selection allows us to train the models in a faster way taking into account the sentences which contain relevant information of the domain and avoids noisy data. We also did a preprocessing to convert html entities and filter out the sentences with more than 50 words for both source and target languages. We finally end up with a selected corpus of 2M sentences (50.5 millions of words), leading to 147K unique tokens for English side and 266K unique tokens for French side.

[3]https://sites.google.com/site/iwsltevaluation2015

[4]https://www.ted.com

3.2. Training

We chose the following hyperparameters to train the systems. The embedding and recurrent layers have a dimension of 620 and 1000, respectively. The model is trained with standard SGD and the minibatch size is set to 80 sentences. The learning rate is updated with the Adadelta method [19]. The norm of the gradient is clipped to be no more than 1 [20] and the weights are initialized with *Xavier* [21]. The validations start at the second epoch and are performed every 5000 updates. Early stopping is based on BLEU [22] with a patience set to 10 (early stopping occurs after 10 evaluations without improvement in BLEU). The vocabulary size of the source and target languages is set to 30K (as the other state of the art models). For the Factored approach, we have 30K vocabulary size for the lemmas and 142 for the factors. This allows the system to possibly generates 172K words with the MACAON tool. Once the model is trained, we set the beam size to 12 (as this is the standard value for NMT [1]) when translating the development corpus. The models were trained during 6 days and translation takes 30 minutes, approximately.

3.3. Feedback

As explained in section 2.1, the RNN decoder is a conditional-GRU. The first GRU cell of the decoder is fed by its previous hidden state and the feedback (*i.e.* the previous generated symbol) with the following formulation.

$$\text{GRU}_1\left(y_{j-1}, \mathbf{s}_{j-1}\right) = (1 - \mathbf{z}_j) \odot \underline{\mathbf{s}}_j + \mathbf{z}_j \odot \mathbf{s}_{j-1}, \quad (1)$$

$$\underline{\mathbf{s}}_j = \tanh\left(\mathbf{W}\mathbf{fb}(y_{j-1}) + \mathbf{r}_j \odot (\mathbf{U}\mathbf{s}_{j-1})\right), \quad (2)$$

$$\mathbf{r}_j = \sigma\left(\mathbf{W}_r\mathbf{fb}(y_{j-1}) + \mathbf{U}_r\mathbf{s}_{j-1}\right), \quad (3)$$

$$\mathbf{z}_j = \sigma\left(\mathbf{W}_z\mathbf{fb}(y_{j-1}) + \mathbf{U}_z\mathbf{s}_{j-1}\right), \quad (4)$$

where $\mathbf{fb}$ is the function which computes the feedback from the previous output y_{j-1}, $\underline{\mathbf{s}}_j$ is the internal representation, $\mathbf{r}_j$ and $\mathbf{z}_j$ being the reset and update gate activations. $\mathbf{W}$, $\mathbf{U}$, $\mathbf{W}_r$, $\mathbf{U}_r$, $\mathbf{W}_z$, $\mathbf{U}_z$ are the trained parameters[5]. Tanh and σ refer to the hyperbolic tangent and the logistic sigmoid activation functions, respectively.

Since we now have two outputs, we need to define what kind of feedback is more suitable for the Factored NMT system. Several solutions are possible to use either or both embeddings as feedback (see Figure 3).

The first assumption we made is highly dependent on the design of the considered factors, *i.e.* the lemmas are the most informative factors among all. Then, we tried using only the lemma embedding as feedback (see equation 5).

$$\mathbf{fb}(y_{t-1}) = y^L_{t-1} \quad (5)$$

where y^L_{t-1} is the embedding of the lemma generated at previous timestep.

[5]All the biases have been omitted for simplicity.

In the same direction but with the other output information, we used only the factors embedding as feedback (see equation 6).

$$\mathbf{fb}(y_{t-1}) = y^F_{t-1} \quad (6)$$

where y^F_{t-1} is the embedding of the factors generated at the previous timestep.

Another straightforward operation is to sum the embeddings (technique used in [23]) of the previous lemma with the embedding of the previous factors, as described in equation 7.

$$\mathbf{fb}(y_{t-1}) = y^L_{t-1} + y^F_{t-1} \quad (7)$$

While this could seem unnatural, by doing this, we hope to obtain a joint vector representation of both the lemma and the factors.

We investigated whether the neural network can learn a better combination of the lemmas and factors embeddings using a linear (eq. 8) or non-linear (eq. 9) operation instead of a simple sum.

$$\mathbf{fb}(y_{t-1}) = (y^L_{t-1} + y^F_{t-1}) \cdot W_{fb} \quad (8)$$

$$\mathbf{fb}(y_{t-1}) = \tanh\left((y^L_{t-1} + y^F_{t-1}) \cdot W_{fb}\right) \quad (9)$$

In this case, W_{fb} are trained weights. In addition, we used the concatenation of both target embeddings as feedback using a linear (eq. 10) or non-linear (eq. 11) operation instead of the sum of them. The concatenation of the embeddings allows us to get full benefit of both outputs for the feedback of the model.

$$\mathbf{fb}(y_{t-1}) = [y^L_{t-1}; y^F_{t-1}] \cdot W_{fb} \quad (10)$$

$$\mathbf{fb}(y_{t-1}) = \tanh\left([y^L_{t-1}; y^F_{t-1}] \cdot W_{fb}\right) \quad (11)$$

Table 1 presents the results obtained with systems integrating the different output embedding combinations as feedback when comparing with the state of the art systems standard NMT, NMT using BPE symbols and multilingual NMT systems.

For sake of comparison, we have computed BLEU at word level using BPE method [7]. We computed the subwords units in the output side of the neural network as done with Factored approach. We set the number of merge operations for the BPE algorithm, as explained in the paper [7], following equation 12.

$$\#merge\ ops = vocab.\ size\ (30K) - \#characters \quad (12)$$

We can see that BPE performs worse (see Table 1) obtaining lower %BLEU than NMT and FNMT systems (lemma and tanh concatenation feedback). BPE system does not generate unknown words because all are encoded as BPE units. Besides this, BPE cannot improve in terms of %BLEU because the generation of BPE units sometimes provides incorrect words when producing the final word level translation.

Moreover, we also compared the FNMT system to the multiway, multilingual NMT system [16]. This method can train several encoder and decoders sharing only the attention mechanism between them. In order to reproduce our experiments using the multilingual architecture, we used one input encoder (*at word-level*) for English and two separate decoders : French lemmas and French factors. The final word is obtained by the factors-to-word process as used with our FNMT system. As presented in Table 1, Multilingual approach performs better than all other systems at lemma and factors level. However, the performance at word level is the lowest due to the desyncronization of the two outputs which are trained independently.

On the other hand, FNMT can generate the words taking into account the relation between lemmas and factors. FNMT system performs better than BPE and multilingual systems but a bit worse than NMT system (-0.13). Nevertheless, the number of unknown words are reduced to more than halved which is not reflected by the automatic score. Section 3.6 shows examples of the FNMT performance managing the unknown words produced by NMT system. Furthermore, we see that the %BLEU score at lemma level performs almost (-0.28) like the lemma %BLEU of multilingual system. On the contrary, the %BLEU at factors level performs much lower (-3.37).

Then, we performed additional experiments to study how we can include the factors embedding for feedback to improve the performance. Firsly, we explored the possibility of having just the factors embedding as feedback. We observe in Table 1 that using factors embedding option, factors-level %BLEU score is the highest in the FNMT systems. By contrast, lemma %BLEU is very low (more than 3 point less) and has a great impact at word level evaluation performing with 3 point less of %BLEU.

Secondly, we performed the sum of the embeddings. The sum of the embeddings without the linear and tanh transformation gives better scores and less number of unknown words than using its linear and tanh operation. In addition, we have experimented the concatenation of the two outputs embeddings to give more information as feedback and learn better the combination of them. The sum and the linear transformation of the embeddings concatenation perform similar in %BLEU in all the levels. By contrast, the tanh transformation of the embeddings concatenation gets an improvement in terms of %BLEU and unknown words comparing to only lemma feedback and linear transformation of the concatenation of the embeddings. This can be due to the fact that we are using a more complete information as feedback and the model can learn how to represent the two embeddings.

3.4. Dependency models to improve factors prediction

We have evaluated the lemma and factors outputs, separately in the standard NMT system, having a different model for each one.

We saw that the lemma score performs similar for FNMT and NMT system. Nevertheless, the difference between the two systems is big (+2.5% BLEU) when evaluating the factors. Moreover, the prediction of factors should be an easy

Model	Feedback	%BLEU word	%BLEU lemma	%BLEU factors	#UNK
NMT	Words	34.69	-	-	1841
BPE	Subwords	34.34	-	-	0
Multilingual	Lemma & Factors	28.70	37.72	45.81	871
FNMT	Lemma	34.56	37.44	42.44	798
FNMT	Factors	31.49	34.05	44.73	821
FNMT	Sum	34.34	37.03	44.16	809
FNMT	Linear Sum	34.27	36.97	44.12	865
FNMT	Tanh Sum	33.92	36.72	43.95	875
FNMT	Linear Concat	34.34	36.97	44.10	836
FNMT	Tanh Concat	34.58	37.32	44.33	792

Table 1: Performance in terms of number of unknown words and %BLEU computed on word, lemma and factors of the state of the art NMT, BPE and multilingual systems and FNMT system when using different output embedding combinations as feedback.

task considering that the output layer size is only 142. In order to compensate for this loss, we have explored different architectures aiming at improving the factors output.

The first experiment we performed to model the dependency consists on a chain of two models with standard NMT system. The first model has as input the source words as usual and as output the target lemmas. Then, the second model translates from the target lemmas from the first model to the target factors. This second model has the restriction of generating the same length sentences for source and target, due to the fact that each lemma has their corresponding factors. Finally, we construct the final words from the output lemmas from the first model and the output factors from the second model.

Model	%BLEU
$EN_{words} - FR_{lemmas}$	37.38
$FR_{lemmas} - FR_{factors}$	90.54
$Factors - to - word$	33.82

Table 2: $EN_{words} - FR_{lemmas} - FR_{lemmas} - FR_{factors}$ chain model results in terms of %BLEU.

Table 2 shows the results of %BLEU of the output of each model step. The first model of the chain performs similar to the FNMT system at lemma-level (comparing with previous Table 1). Moreover, the score of the output of the second model is very high due to the easy task of translating from lemma to factors on the same language. Nevertheless, factors-to-word process to build the words from the outputs of the two models, obtains worse score than NMT and FNMT models (see Table 1 for comparison). This can be due to the difficulty of factors-to-word process handling the asynchronous outputs of both models trained separately, and having different alignments between the source and target words. This experiment gives us an idea of creating a dependency from lemmas to factors, due to the high performance, to help the factors output to produce higher %BLEU score.

3.4.1. Lemma dependency

One observation that can be made is that while generating factors could seem easier due to the small number of the possible outputs (only 142), the BLEU score is not as high as what we could expect. However, one could argue that generating a sequence of factors in French from a sequence of English words is not an easy task. In order to help the factors prediction, we contextualized the corresponding output with the lemma being generated. This creates a dependency between the lemma output and the factors output. We built models, where the factors output is directly dependent of the lemma, in order to receive more information of it as it has the main information of the word. The dependency has been implemented by including an extra input (see Figure 4 left side) which projects the lemma embeddings into the hidden layer used to generate factors. We use as main feedback only the previous lemma.

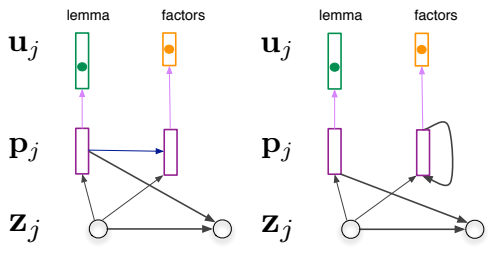

Figure 4: Factors prediction dependency models decoder detail: lemma dependency (left) and factors dependency (right).

We have implemented two possibilities for the lemma dependency model to give as additional input to the factors output:

1. The previous generated lemma to give it directly as recent context.

2. The current generated lemma corresponding to the factors to be generated.

3.4.2. Factors dependency

Another architecture has been implemented to improve the factors output performance. In order to take advantage of the information of the previous generated factors, we use them as feedback to the factors generation (see Figure 4 right side).

		%BLEU			
Model	**Depend.**	**word**	**lemma**	**factors**	**#UNK**
NMT	-	34.69	-	-	1841
FNMT	-	34.56	37.44	42.44	798
FNMT	prev. lem.	34.34	37.39	42.33	831
FNMT	curr. lem.	34.62	37.30	43.36	690
FNMT	prev. fact.	**34.72**	37.56	43.09	794

Table 3: Dependency different options of models results interms of %BLEU at word, lemma and factors level and the number of unknown words.

The results applying the dependency models techniques are presented in Table 3. We can observe that the dependency model using the previous lemma does not improve the results respect to the FNMT model without dependency. This can be due to the fact that we have already given the previous lemma as main feedback for the recurrent hidden state. On the other hand, the dependency model using the current lemma improves (+0.06) the performance of the FNMT without dependency in terms of %BLEU score on words. If we see the results of the lemma and factors level BLEU, we obtained 0.14 lower %BLEU at lemmas level when using this dependency. By contrast, the factors output obtains an improvement of almost one point of %BLEU. Moreover, the number of unknown words is the lowest value compared to the rest of the models. This model allows us to improve and specialize the produced factors with the generated lemma. However, it has an impact at lemmas output which is more correlated to the word evaluation.

The last model, using dependency with the previous factors to feed the factors output, increases the %BLEU value when comparing with the NMT and FNMT other systems. We can also observe that the %BLEU at lemma and factors level has improved. Furthermore, the number of unknown words has decreased by 4. This architecture has improvements due to the benefit of using all the previous information (lemma and factors) and input the previous factors in the specialized layer of this output.

3.5. Translating long sentences

In this section, we compare the results of the standard NMT system with the FNMT system when translating long sentences. For this, we observed the translation performance with respect to source sentences length. Figure 5 shows the %BLEU score for sentences between 10 and 100 words with intervals of 10.

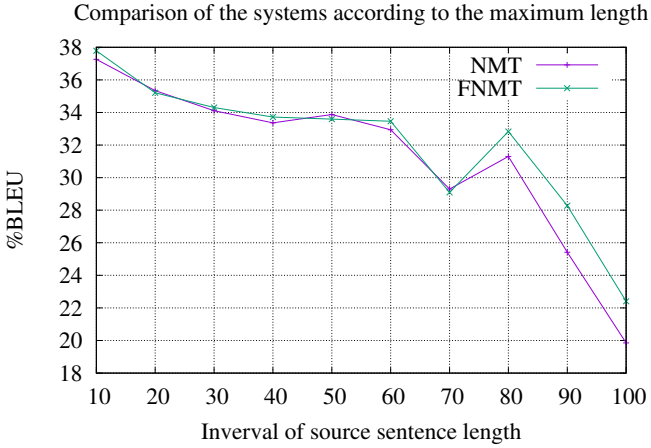

Figure 5: Comparison in terms of %BLEU of the NMT and FNMT with factors dependency systems according to the maximum source sentence length.

We can observe that FNMT system performs similar than NMT system in the intervals smaller than 80. By contrast, we can see that the FNMT system helps significantly with translating long sentences (between +1.5-3 more points of %BLEU). This improvement can be due to the expressiveness of the FNMT system generating new vocabulary.

3.6. Qualitative analysis

FNMT system generates much less unknown words than NMT system because it can cover more vocabulary. Then, we have observed some examples to confirm that the translation of the unknown words are correct and show the performance in a qualitative analysis

3.6.1. Translation examples of FNMT system performance

The translation examples in Table 4 show the FNMT system performance against the NMT system.

First example shows when Factored system can generate words when the NMT base system predicts unknown words. Firstly, the word ‘lineage’ in source sentence is translated as the reference ‘lignée’ by the FNMT system and mapped to ‘UNK’ by the NMT base system. Secondly, the word ‘adaptive’ is translated as ‘adaptatifs’ by the FNMT system, the reference translation is ‘adaptés’, but we can consider the FNMT choice a better translation. NMT system also mapped the word ‘adaptive’ to ‘UNK’. Consequently, BLEU

Words	Src	no one knows what the hell we do								
Words	Ref	personne	ne	sait	ce	que	nous	faisons	.	
Words	FNMT	personne	ne	sait	ce	qu'	être	l'	enfer	.
Lemmas		personne	ne	savoir	ce	qu'	être	l'	enfer	.
Factors		pro-s	advneg	v-P-3-s	prep	prorel	cln-3-s	det	nc-m-s	poncts
Words	FNMT dependency	personne	ne	sait	ce	que	nous	faisons		.
Lemma		personne	ne	savoir	ce	que	nous	faire		.
Factors		nc-f-s	advneg	v-P-3-s	det	prorel	cln-1-p	v-P-1-p		poncts

Table 5: Examples of translations with FNMT and FNMT with dependency.

1	Src	set of adaptive choices that our lineage made
	Ref	de choix adaptés établis par notre lignée
	NMT	de choix UNK que notre UNK a fait
	FNMT	de choix adaptatifs que notre lignée a fait
2	Src	enzymes that repair them and put them together
	Ref	enzymes qui les réparent et les assemblent .
	NMT	enzymes qui les UNK et les UNK .
	FNMT	enzymes qui les réparent et les mettent ensemble.
3	Src	santa marta in north colombia
	Ref	santa marta au nord de la colombie
	NMT	santa UNK dans le nord de la colombie
	FNMT	santa marta dans le nord de la colombie

Table 4: Examples of translations with NMT and Factored NMT.

score penalizes this example in FNMT system being a correct translation.

In the second example, an NMT system translation has generated two unknown words. By contrast, FNMT system can generate correctly the two words producing 'réparent' and 'mettent ensemble'. This is due, on one hand, because the word 'réparent' appears 439 times in the word vocabulary of the NMT system so it is not sufficient frequency to include it in the shortlist. On the other hand, the lemma 'réparer' appears 8523 times in the lemmas shortlist so it is included and we are able to generate 'réparent' from the lemma and factors outputs (verb in present and third person in plural). Moreover, the verb in English 'put together' is translated to 'assamblent' in the reference which is a synonym of the FNMT system translation 'mettent ensemble'. Unfortunately, this is not well measured by %BLEU score.

Example 3 shows an FNMT system translation performing as the reference. We are able to generate the name entity 'marta' that it is not in the shortlist of the NMT system vocabulary. These examples show the potential of the FNMT system generating new words and reducing unknown words.

3.6.2. Dependency model comparison

We have compared the benefits of having the dependency in FNMT system to help the generation of better factors output.

Table 5 shows one example where the FNMT without dependency generates a wrong factors output. Simple FNMT produces the lemma 'être' and its corresponding factors are cln-3-s (nominative clitique, third person of the singular), which are not correct. On the contrary, FNMT with dependency produces correctly, the lemmas 'nous faire' with their factors 'cln-1-p' (nominative clitique, first person of the plural) and 'v-P-1-p' (verb in present, first person of the plural), respectively. This example shows the effectiveness of the dependency method.

4. Conclusions

In this paper, we have proposed an NMT architecture which produces a factored representation of the target language words. Those factors are based on linguistics *a priori* knowledge. We have compared the Factored NMT system with other state of the art systems such as the subwords units and the multiway, multilingual NMT. We have explored different architectures of the Factored NMT system with the different options of feedback and adding a dependency in one of the outputs of the neural network. We showed that we are able to train Factored NMT systems with better performance than the state of the art systems. Hence, with the FNMT system we are able to substantially reduce the generation of unknown words. Also, the use of additional linguistic resources allows us to generate new word forms that would not be included in the standard NMT system with *shortlist*.

For future work, we would like to include linguistic features at the source language. It is known that this can be helpful for NMT [15]. Extending the approach with input factors could make the target language factors generation better. Furthermore, different attention mechanisms for each output will be explored because they could be aligned to different source words. The proposed FNMT architecture could even show better performance if applied when we translate to highly inflected languages like German, Arabic, Czech or Russian. Finally, FNMT approach will be explored in multimodal and multilingual tasks.

5. Acknowledgements

This work was partially funded by the French National Research Agency (ANR) through the CHIST-ERA M2CR project, under the contract number ANR-15-CHR2-0006-01.

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.

[2] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *CoRR*, vol. abs/1409.3215, 2014.

[3] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst, "Moses: Open source toolkit for statistical machine translation," in *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ser. ACL '07. Stroudsburg, PA, USA: Association for Computational Linguistics, 2007, pp. 177–180.

[4] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *CoRR*, vol. abs/1406.1078, 2014.

[5] S. Jean, K. Cho, R. Memisevic, and Y. Bengio, "On using very large target vocabulary for neural machine translation," *CoRR*, vol. abs/1412.2007, 2014.

[6] H.-S. Le, I. Oparin, A. Messaoudi, A. Allauzen, J.-L. Gauvain, and F. Yvon, "Large vocabulary SOUL neural network language models," in *INTERSPEECH*, 2011. [Online]. Available: sources/Le11large.pdf

[7] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," *CoRR*, vol. abs/1508.07909, 2015.

[8] J. Chung, K. Cho, and Y. Bengio, "A character-level decoder without explicit segmentation for neural machine translation," *CoRR*, vol. abs/1603.06147, 2016.

[9] W. Ling, I. Trancoso, C. Dyer, and A. W. Black, "Character-based neural machine translation," *CoRR*, vol. abs/1511.04586, 2015.

[10] M. R. Costa-Jussà and J. A. R. Fonollosa, "Character-based neural machine translation," *CoRR*, vol. abs/1603.00810, 2016.

[11] J. A. Bilmes and K. Kirchhoff, "Factored language models and generalized parallel backoff," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: Companion Volume of the Proceedings of HLT-NAACL 2003–short Papers - Volume 2*, ser. NAACL-Short '03. Stroudsburg, PA, USA: Association for Computational Linguistics, 2003, pp. 4–6.

[12] 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*. Berlin, Germany: Association for Computational Linguistics, August 2016, pp. 74–82.

[13] A. Alexandrescu, "Factored neural language models," in *In HLT-NAACL*, 2006.

[14] Y. Wu, H. Yamamoto, X. Lu, S. Matsuda, C. Hori, and H. Kashioka, "Factored recurrent neural network language model in ted lecture transcription," in *IWSLT*, 2012.

[15] R. Sennrich and B. Haddow, "Linguistic input features improve neural machine translation," *CoRR*, vol. abs/1606.02892, 2016.

[16] 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

[17] A. Nasr, F. Béchet, J.-F. Rey, B. Favre, and J. L. Roux, "Macaon, an nlp tool suite for processing word lattices," in *Proceedings of the ACL-HLT 2011 System Demonstrations*, 2011, pp. 86–91.

[18] A. Rousseau, "XenC: An open-source tool for data selection in natural language processing," *The Prague Bulletin of Mathematical Linguistics*, vol. 100, pp. 73–82, 2013.

[19] M. D. Zeiler, "ADADELTA: an adaptive learning rate method," *CoRR*, vol. abs/1212.5701, 2012.

[20] R. Pascanu, T. Mikolov, and Y. Bengio, "Understanding the exploding gradient problem," *CoRR*, vol. abs/1211.5063, 2012.

[21] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10). Society for Artificial Intelligence and Statistics*, 2010.

[22] 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*, ser. ACL '02, Stroudsburg, PA, USA, 2002, pp. 311–318.

[23] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," in *Proceedings of Workshop at ICLR*, 2013.