Factored neural machine translation

Mercedes García Martínez

UNIVERSITE BRETAGNE LOIRE

Le Mans Université

Thèse de Doctorat

Mercedes
GARCÍA MARTÍNEZ

Mémoire présenté en vue de l'obtention du
grade de Docteur de Le Mans Université
sous le sous le sceau de l'Université Bretagne Loire

École doctorale : MathSTIC

Discipline : Informatique
Spécialité : Informatique
Unité de recherche : LIUM

Soutenue le 27 mars 2018
Thèse N° : 2018LEMA1002

Factored Neural Machine Translation

JURY

Rapporteurs :	**Marcello FEDERICO**, Professeur, Fondazione Bruno Kessler, Trento **Alexandre ALLAUZEN**, Maître de conférence, LIMSI-CNRS, Université Paris-Sud
Examinateurs :	**Lucia SPECIA,** Professeur, University of Sheffield **Francisco CASACUBERTA,** Professeur, Universitat Politècnica de València
Directeur de Thèse :	**Yannick ESTÈVE**, Professeur, LIUM, Le Mans Université
Co-directeur de Thèse:	**Loïc BARRAULT**, Maître de conférence, LIUM, Le Mans Université
Co-encadrant de Thèse :	**Fethi BOUGARES**, Maître de conférence, LIUM, Le Mans Université

Acknowledgments

The period doing my PhD has been hard but really rewarding. First and foremost, I would like to thank my supervisor, Loïc Barrault, without him it would have been impossible to do my PhD. I extend my appreciation to my other two supervisors: Fethi Bougares has helped me a lot those years to reach my achievements and Yannick Estève has always supported me in my work.

I would also like to thank the other members of my thesis committee, Alexandre Allauzen, Marcello Federico and Lucia Specia for giving me feedback and making sure I can graduate. I extend my gratitude to Nathalie Camelin to evaluate my thesis every year and Dominique Py to help me with the administration of my PhD paperwork.

I am thankful to my colleges of the translation group in my lab, Walid Aransa, Ozan Caglayan, Adrien Bardet and Haithem Afli. Specially, Walid Aransa for encouraging me in the difficult times and for all his help.

All these years in France would not be good without my colleges in the lab and great people I met in Le Mans. They taught me their beautiful language and showed me their big hearth. Above all, Sahar Ghannay for encouraging me to fight for my PhD and Mélanie Hardy for being with me day by day.

The main support in my life are my close friends and my family. They tell me what I need to hear, not what I want to hear. They are always with me in the good and bad times. Without them I have no idea where I would be. Firstly, thank you to Prashansa Dhawan for her great help with my English and her very good advises. Also, I want to say thank you to my also expatriate friend Alberto Muñoz Arancón to be my biggest support in France together with María Ramiro Martín for the great times in Le Mans. My group of music "Takasouffler" helped me a lot those years, thank you Jean Marc, María and Vicent. Thank you as well to Geneviève and Philippe for all your help in Le Mans. I do not want to forget Barto Mesa Lao to help me to find a PhD position and give me some guidelines to achieve my thesis. I am very grateful to my group of friends in Valencia "los molones" and my friends I met in my wonderful summers in Carboneras. Finally but not least, my family is the most important thing in my life. Without the unconditional support of my parents, brothers, sister, brother in law, sisters in law and my lovely nephews and niece, this thesis would not be a reality. Lastly, I dedicate this thesis to my grandmother who I will always remember.

Dedicated to my family.
Dedicado a mi familia.

Résumé

La diversité des langues complexifie la tâche de communication entre les humains à travers les différentes cultures. La traduction automatique est un moyen rapide et peu coûteux pour simplifier la communication interculturelle. Récemment, la Traduction Automatique Neuronale (NMT) a atteint des résultats impressionnants. Cette thèse s'intéresse à la Traduction Automatique Neuronale Factorisé (FNMT) qui repose sur l'idée d'utiliser la morphologie et la décomposition grammaticale des mots (lemmes et facteurs linguistiques) dans la langue cible. Cette architecture aborde deux défis bien connus auxquelles les systèmes NMT font face. Premièrement, la limitation de la taille du vocabulaire cible, conséquence de la fonction softmax, qui nécessite un calcul coûteux à la couche de sortie du réseau neuronale, conduisant à un taux élevé de mots inconnus. Deuxièmement, le manque de données adéquates lorsque nous sommes confrontés à un domaine spécifique ou une langue morphologiquement riche. Avec l'architecture FNMT, toutes les inflexions des mots sont prises en compte et un vocabulaire plus grand est modélisé tout en gardant un coût de calcul similaire. De plus, de nouveaux mots non rencontrés dans les données d'entraînement peuvent être générés. Dans ce travail, j'ai développé différentes architectures FNMT en utilisant diverses dépendances entre les lemmes et les facteurs. En outre, j'ai amélioré la représentation de la langue source avec des facteurs. Le modèle FNMT est évalué sur différentes langues dont les plus riches morphologiquement. Les modèles à l'état de l'art, dont certains utilisant le Byte Pair Encoding (BPE) sont comparés avec le modèle FNMT en utilisant des données d'entraînement de petite et de grande taille. Nous avons constaté que les modèles utilisant les facteurs sont plus robustes aux conditions d'entraînement avec des faibles ressources. Le FNMT a été combiné avec des unités BPE permettant une amélioration par rapport au modèle FNMT entrainer avec des données volumineuses. Nous avons expérimenté avec différents domaines et nous avons montré des améliorations en utilisant les modèles FNMT. De plus, la justesse de la morphologie est mesurée à l'aide d'un ensemble de tests spéciaux montrant l'avantage de modéliser explicitement la morphologie de la cible. Notre travail montre les bienfaits de l'application de facteurs linguistiques dans le NMT.

Abstract

Communication between humans across the lands is difficult due to the diversity of languages. Machine translation is a quick and cheap way to make translation accessible to everyone. Recently, Neural Machine Translation (NMT) has achieved impressive results. This thesis is focus on the Factored Neural Machine Translation (FNMT) approach which is founded on the idea of using the morphological and grammatical decomposition of the words (lemmas and linguistic factors) in the target language. This architecture addresses two well-known challenges occurring in NMT. Firstly, the limitation on the target vocabulary size which is a consequence of the computationally expensive softmax function at the output layer of the network, leading to a high rate of unknown words. Secondly, data sparsity which is arising when we face a specific domain or a morphologically rich language. With FNMT, all the inflections of the words are supported and larger vocabulary is modelled with similar computational cost. Moreover, new words not included in the training dataset can be generated. In this work, I developed different FNMT architectures using various dependencies between lemmas and factors. In addition, I enhanced the source language side also with factors. The FNMT model is evaluated on various languages including morphologically rich ones. State of the art models, some using Byte Pair Encoding (BPE) are compared to the FNMT model using small and big training datasets. We found out that factored models are more robust in low resource conditions. FNMT has been combined with BPE units performing better than pure FNMT model when trained with big data. We experimented with different domains obtaining improvements with the FNMT models. Furthermore, the morphology of the translations is measured using a special test suite showing the importance of explicitly modeling the target morphology. Our work shows the benefits of applying linguistic factors in NMT.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACRONYMS

MT	Machine Translation
NMT	Neural Machine Translation
FNMT	Factored Neural Machine Translation
IFNMT	Interleaved Factored Neural Machine Translation
BPE	Byte Pair Encoding
FBPE	Factored Byte Pair Encoding
SMT	Statistical Machine Translation
PBMT	Phrase-Based Machine Translation
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
GRU	Gated Recurrent Units
LSTM	Long Short-Term Memory
LM	Language Model
NLM	Neural Language Model
WER	Word Error Rate
BLEU	Bilingual Evaluation Understudy
BEER	BEtter Evaluation as Ranking
TER	Translation Edit Rate
CTER	Translation Edit Rate on Character-Level
PoS	Part of Speech
`UNK`	Unknown word
UR	`UNK` replacement
h2o	hidden to output
fb	feedback
IWSLT	International Workshop of Spoken Language Translation
TED	Technology Entertainment Design
WMT	Workshop on Machine Translation
DCEP	Digital Corpus of the European Parliament
LETA	Latvian information agency
EU	European Union
DA	Direct Assessment
GPU	Graphics Processing Unit
SOUL	Structured Output Layer
CCG	Combinatorial Categorial Grammar
CRF	Conditional Random Field
CONDOR	COnstrained, Non-linear, Direct, parallel Optimization using trust Region
dep	dependent
ind	independent

CHAPTER 1

INTRODUCTION

Contents

"There can never be an absolutely final translation".
Robert M. Grant

1.1 Machine Translation

History has shown the need for a continuous exchange of culture and products across the land. Human beings have to communicate with each other in order to trade and socialise. The main challenge is to be able to easily communicate considering the diversity of languages and their differences. There are around 7000 languages in the world[1]. Translation breaks language barrier, which separates people.

Nowadays, translation plays an essential role in the economical, cultural and social development of countries. The necessity of fast, high quality translation services are becoming more important. This is motivated by the importation and exportation of products and knowledge to create stronger relations between countries. Moreover, translation plays a fundamental role in the increase of international tourism and the diffusion of media through television and the Internet.

Machine translation (MT) is a quick and cheap way to make translation accessible to everyone. It consists of automatically translating human languages using

[1] LangScape: The Language of Landscape: Reading the Anglo-Saxon Countryside. <http://langscape.org.uk>, version 0.9, accessed 1 November, 2008.

computers. Knowing the importance of translation, research on machine translation has a long history since computers were invented and it remains an important and still unsolved topic.

The first MT systems were rule-based built only using linguistic information (Nirenburg, 1989). The translation rules were manually created by experts. Although the rules are well defined, this process is very expensive and does not generalize well to all domains and languages.

Statistical Machine Translation

Afterwards, statistical machine translation (SMT) started to outperfom rule-based systems. SMT systems automatically create the translation rules using parallel corpora. Handcrafted rules or linguistic knowledge are not required to build an SMT system. This fact makes SMT cheaper and quicker to perform but it lacks of human supervision.

Formally, the goal of SMT is to find the most probable target sequence ($\hat{f}$) given a source (e) sequence.

$$\hat{f} = \underset{f}{\operatorname{argmax}}\, p(f|e) \tag{1.1}$$

Equation 1.1 defines the SMT decoder.

From the Bayes theorem we obtain Equation 1.2.

$$p(f|e) = \frac{p(e|f)p(f)}{p(e)} \tag{1.2}$$

Because e is fixed for all f, the maximization over f does not depend on e, thus, Equation 1.1 is equivalent to Equation 1.3 which is the Bayesian noisy channel.

$$\hat{f} = \underset{f}{\operatorname{argmax}}\, p(f|e) = \underset{f}{\operatorname{argmax}}\, p(e|f)p(f) \tag{1.3}$$

The probability $p(e|f)$ defines the translation model and the probability $p(f)$ is provided by the language model (LM). The LM is an important component in SMT. It was previously used in speech recognition (Jelinek et al., 1975) and several other NLP tasks. The LM is trained with a monolingual corpus in the target language. The probability distribution over a sequence of symbols, often words, is specified by the LM.

More details about LM calculation are defined in Equation 1.4. The LM decomposes the probability of a word sequence $f = f_1, \ldots, f_m$ as follows:

$$p(f) = \prod_{i=1}^{m} p(f_i|f_{<i}) \tag{1.4}$$

where each of the individual terms $p(f_i|f_{<i})$ are the conditional probability of the generated current word f_i given the previous words or context $f_{<i}$.

Having the LM, we can know if a sequence of words is more likely to appear than another.

The first SMT approach was taking the word as a translation unit (Brown et al., 1993). Later, phrased-based models were invented (Koehn et al., 2003) translating several contiguous words which permit more context and a better quality of the automatic translation. Phrase-based machine translation (PBMT) also incorporates the alignment ability (Och and Ney, 2004) between words. The alignment in both directions of language pairs obtains many-to-many correspondences of subparts of the sentences, which are called *phrases*. Those phrases are stored in a phrase table with their corresponding scores. The scores kept in the phrase table contain information about the direct and inverse phrase translation probability and direct and inverse lexical probabilities.

The most popular tool used for SMT is Moses (Koehn et al., 2007). The SMT approach uses a log linear model determined by features functions (see Equation 1.5). Those features come from the information about language model, translation model, reversed translation model, distance-based reordering model, lexicalized reordering model and length/unknown penalties, principally.

$$\log p(f|e) = \sum_{i=1}^{N} \lambda_i \log h_i(f, e) \tag{1.5}$$

where N represents the number of features and λ_i the feature weight of h_i which is the $i-th$ feature function.

The log-linear model makes possible, the combination of several components to determine the quality of the translation. Each component is represented by one or more features, which are weighted, and summed together. In order to tune the model for decoding, an additional test set is evaluated to set the weights.

Another approach is the hierarchical phrase-based model (Chiang, 2005) which extends the phrase-based approach with context-free grammar learning without the need of linguistic information. This technique is specially used for language pairs like English-Chinese for which a reordering of the sentences will help to make a better translation due to the distinct grammar of the two languages. For example, a hierarchical rule translation can model the negation in French to distinguish it from the negation in English (see example in Equation 1.6).

$$(ne \quad X_1 \quad pas \quad , \quad do \quad not \quad X_1) \tag{1.6}$$

There are also other works about adding syntax to SMT (Wu, 1997; Yamada and Knight, 2001; Chiang, 2005). Those approaches are shown to improve the translation quality, specially in languages with different structure such as Chinese and English. However, they imply the need of linguistic information.

Despite the improvement reached by using classical SMT models, the context handled by the LMs models is short and thus its generalization ability is limited. The possible n-grams are stored on big tables together with their probabilities. Due to data sparsity, probabilities are not well estimated for big n-gram length because

they are not frequent in the corpora. Therefore, the context is limited to 4 or 6 n-grams. Neural language models (NLMs) (Bengio et al., 2003) use a continuous word representation where similar words are close in the projection space. NLMs do not need to store the n-grams and can work with longer context using feed-forward or recurrent neural networks. After the development of the NLMs, SMT started to benefit from them (Schwenk, 2010; Le et al., 2012). This benefit is obtained by adding the NLM probability to the n-best list generated during decoding. Then, the n-best list is rescored achieving a better translation quality. More recently, NLMs have been improved adding the source words jointly with the target context (Schwenk, 2012; Devlin et al., 2014). This last component makes the SMT full pipeline very complex. We find two main drawbacks in SMT: (1) the complexity of the model increases adding more and more features to the loglinear framework (Green et al., 2013) and (2) as we translate phrase-by-phrase, the context taken into account is short and consequently, long-distance dependencies are not captured. Because of that, the machine translation process was re-designed, in order to address those inconveniences using neural networks.

Neural Machine Translation

Neural Machine Translation (NMT) went from a fringe research activity in 2014 to the widely-adopted leading way to do MT in 2016. NMT consists of a big neural network with millions of weights that models all the MT pipeline. Parallel corpus is still required to train the NMT model but the preprocessing steps are simpler. NMT is trained in an end-to-end manner without the need of several components. Unlike the classical phrase-based translation which handles the translation at phrase level with a limited context, NMT is able to manage long-range dependencies. An encoder represents the entire source sentence and a decoder processes this information to generate the translation. This fact makes the NMT models very powerful since each decoding step is conditioned by a context vector of the whole source sentence. In the last years, NMT systems started to outperform SMT systems in the MT evaluation campaigns and it is becoming to be used by the translation industry. All the work in this thesis is done in the NMT framework.

In recent years, NMT has developed very quickly. The first machine translation models using neural networks appeared some decades ago (Chrisman, 1991; Neco and Forcada, 1997; Castaño and Casacuberta., 1997; Allen, 1987) but the hardware at that moment was not enough developed to get good results. Nowadays, the new GPU processors capable of executing many operations in parallel have made it possible for the neural network power to be a reality in machine translation.

The first work to map the input sentence into a vector and then back to a sentence was done by Kalchbrenner and Blunsom (2013). Afterwards, the sequence to sequence encoder-decoder NMT model was introduced (Cho et al., 2014b; Sutskever et al., 2014). The sequence to sequence encoder-decoder NMT model is based on two recurrent neural networks (RNN), one used for the encoder and the other for the decoder. A detailed description of RNNs is given in Chapter 2.

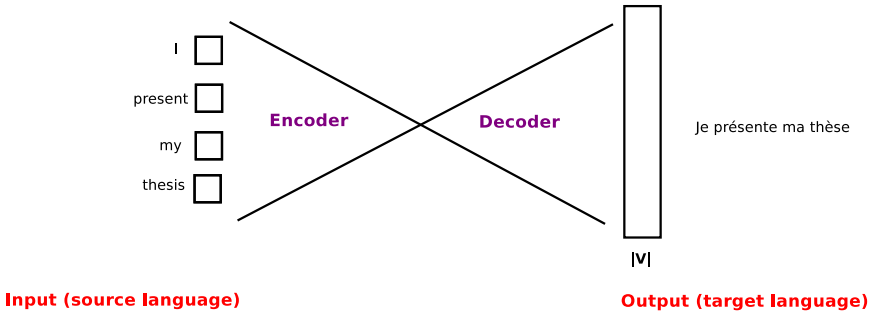

Figure 1.1: Sequence to sequence NMT architecture.

The source language sequence is embedded in the encoder and the decoder maps the representation back to a target language sequence (see Figure 1.1). The encoder and the decoder are trained jointly to maximize the conditional probability of the reference translation. More formally, NMT maximizes the conditional probability $p(f|e)$ where e is the source sequence and f is the target sequence (see Equation 1.7).

$$p(f|e) = \sum_{t=1}^{T} \log p(f_t|f_{<t}, \mathbf{e}) \tag{1.7}$$

The encoder creates the representation of the source $\mathbf{e}$ and the decoder generates the translation sentence, one token at each timestep (t) based on the conditional probability of the previous generated tokens and the source representation. Each generated token is selected based on the calculated probability from the output layer, which has the size of vocabulary.

The relevant information from the source sentence has to be kept by the encoder-decoder model, this is difficult when dealing with long sentences. One first approach to resolve the performance drop with long sentences consists of reversing the tokens of the source sentence starting from the last one and finishing by the first one (Sutskever et al., 2014). This approach reduces the distance between the first words of the source sequence and the first words in the target sequence which helps to improve the translation performance.

Later, Bahdanau et al. (2014) introduced an attention mechanism into the NMT with huge success. Nowadays, NMT with attention mechanism systems are the state of the art.

1.2 Thesis Outline

Machine translation is a complex task and there is still a lot of work ahead to improve it. We can enumerate some current hot topics in NMT.

1. **Data sparsity**: MT systems often only rely on the training corpora which are always available in limited quantity and often difficult to find. Bilingual corpora are expensive to build and not available for some specific language

pairs or domains. The translation of a highly inflected languages is even more difficult because not enough samples with all the possible inflected forms are seen in the training set. This is a challenge which remains open on the SMT systems.

2. **Vocabulary size limitation**: due to the computational complexity of the output layer, the target vocabulary size is often limited. Therefore, it is not possible to generate all the words seen in the training dataset. This can lead to the generation of unknown words that are not included in the vocabulary.

NMT systems often do not incorporate any additional linguistic information, they only rely on the training dataset. Linguistic motivated systems may help to overcome data sparsity, generalize and disambiguate to finally improve translation facing the previously described challenges. The integration of linguistics in an NMT system is not a trivial task. Usually, linguistic tools are not specifically developed to be integrated with NMT systems. Moreover, they could produce some errors since they are generally trained with machine learning techniques using annotated corpora.

Factored NMT systems are introduced in this thesis with the aim of integrating linguistics in NMT systems. We focus on the inflection morphological phenomena. In this thesis, I propose a new architecture that provides the possibility of including linguistic information in order to predict a better translation. This architecture can also be used for other tasks where additional information is required to train the network.

This document is organized in two parts. The first part is about the formal background work and state of the art techniques. Chapter 1 focuses on the history of machine translation explaining briefly the previous approaches and introducing the recent research of NMT. In Chapter 2 - *Background*, the readers find the principles of NMT. Section 2.1 is about recurrent neural networks, which are the basic components of the NMT system used for this thesis, including the backpropagation algorithm for training and some settings of the network. Section 2.2 describes the baseline system used for this thesis, it consists of an NMT system with an attention mechanism.

Chapter 3 - *State of the art* is a review of previous related works clustered in three sections: approaches which also deal with large vocabulary (Section 3.1), factors related approaches (Section 3.2) and other works which also handle multiple inputs and outputs in a neural network (Section 3.3).

The second part includes my contributions which is composed of 3 chapters. Chapter 4 - *Factored Neural Machine Translation* contains the description of the Factored NMT approach and a set of experiments with it. The three sections contain the set of experiments: (1) Preliminary experiments to test the FNMT system in Section 4.3, (2) experiments to design the FNMT architecture options in Section 4.4 and (3) further experiments including more morphology and the incorporation of factors at input side of the network in Section 4.5. Next Chapter 5 -

Translating Morphologically Rich Languages is about translation of morphologically rich languages using Factored NMT system. English to Czech and English to Latvian WMT'17 data have been used to performed the experiments in Section 5.1. Section 5.2 contains a study of the impact of the data for Arabic to French translation. A qualitative evaluation of the translations has been carried out in Section 5.3. Lastly, the conclusions and perspectives are presented in Chapter 6.

Part I

Background and State Of the Art

CHAPTER 2

BACKGROUND

Contents

"Every language is a world. Without translation, we would inhabit parishes bordering on silence". George Steiner

In this chapter, we will describe first the concept of recurrent neural networks and we will continue explaining the different methodologies used to build a neural machine translation model.

2.1 Recurrent Neural Networks

In machine learning, artificial neural networks are a family of models inspired by biological neural networks. From this point on, we will refer to the artificial neural network using only neural networks. Neural networks are used to estimate or approximate functions that can depend on a large number of inputs. They are generally presented as systems of interconnected weights that can be tuned on a dataset with samples. Their structure is based on input layers, hidden layers and output layers.

In our context, we need to convert words into numeric vectors. A straightforward way of doing this consists of using a *one-hot* representation. This method converts the word into a sparse representation with only one element of the vector set to 1, the rest being zero. By means of this method, the input vocabulary is projected in a hidden layer learning a continuous representation of the words. The weights connecting this layer are the new word vectors. The weight matrix essentially becomes a look-up or encoding table of the words. These weight values contain context information. This process is called *word embedding*.

A recurrent neural network (RNN) is a neural network where some connections form a directed cycle, giving feedback information to the next state of learning. RNNs models are specially useful for modeling sequential data like sequences of words. This is the case of the machine translation or language modeling tasks where the word sequences have unbounded and different length. The RNN created by Elman (1990) uses as cycles the context extracted from the weights between source and target (hidden layer).

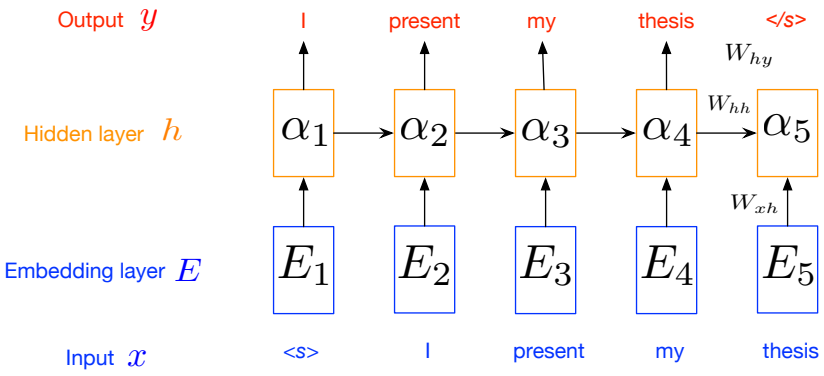

Figure 2.1: Example of RNN for language modeling task.

Figure 2.1 shows an example of RNN when applying to language modeling task. The input of an RNN is a sequence of vectors $x_1, x_2, \ldots, x_n$ representing a sentence or n-gram. The embedding layer E_i learns weights to represent a word. Each new input x_i is processed and the memory of the network is updated producing a hidden state h_i. The recurrence of the network consists of the update of each hidden state.

This update is processed with a function computing the current input $\mathbf{x}_t$ and the previous hidden state h_{t-1}. The first state h_0 is set with an initialization function (σ) which can be a random value or 0.

Equation 2.1 shows more details about the function to update the hidden states h_t at each timestep. This function often consists of a non-linear function (f) such as logistic (*sigmoid*) or hyperbolic tangent (*tanh*).

$$h_t = f(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}h_{t-1}) \tag{2.1}$$

where each $\mathbf{W}_{xh}$ represents a weights matrix connecting the input sequence and the hidden layer and $\mathbf{W}_{hh}$ a weights matrix connecting the hidden layers in the network (see Figure 2.1).

At each timestep t, an output symbol y_t is generated which is calculated by multiplying the weights $\mathbf{W}_{hy}$ (hidden to target weights) and hidden values h_t (see Equation 2.2).

$$y_t = \mathbf{W}_{hy}h_t \tag{2.2}$$

Another type of RNN is the one created by Jordan (1986) where the cycles are fed with the previous generated timestep (output layer), this is what we call feedback of the model. In this thesis, we use both types of RNN.

The output symbol is decided through the calculation of a softmax function (see Equation 2.3). The softmax function transforms the output scores (y_t) to a probability vector p_t. This p_t vector has the size of the target vocabulary. In order to calculate the output value, at each timestep, a softmax function is required where each value is divided by the sum over all values. The softmax normalization is computationally expensive. The word with the highest probability is selected as output.

$$p_t = \mathsf{e}^{y_t} / \sum_{r=1}^{N} \mathsf{e}^{y_r} \text{ for } t \in \{1, \ldots, N\} \tag{2.3}$$

where y_t is the output vector at timestep t and p_t the softmax normalization over the total number of outputs N.

In machine translation or language modeling, the vocabulary is often very large. Therefore, the computation of the softmax function is time consuming. Moreover, very large vocabularies cannot fit in memory. This is one of the major computational bottleneck in this field.

2.1.1 Training

Training neural networks requires the optimization of the weights to predict the corresponding output of the training samples. The training method used in this thesis is called backpropagation. For more details about backpropagation method, I address the reader to Li et al. (2012). The samples are fed to the network in batches, repeatedly, until the model is trained. Using batches instead of one example or the

full set of examples allows the system to converge faster and compute more efficiently on GPUs.

During training, the outputs generated by the network are compared to the correct outputs which allows us to compute a cost in order to measure how good is the network. Normally, the system does several passes (epochs) over the training data.

Cost function

In the context of machine translation, the cost function corresponds to the negative log-likelihood of the sentences seen in the training data (see Equation 2.4).

$$C(\theta) = \sum_{(x,y)\in D} -\log p(y|x,\theta) \tag{2.4}$$

where D refers to the bitext (x, y), x is the source sentence, y its corresponding target sentence and θ all the parameters of the neural network model.

Ideally, the cost would be 0 when the correct word probability is 1. This is typically an approximation, hence we have higher cost than 0. The overall sentence cost is the sum of all word costs.

Stochastic Gradient Descent

The optimal weights are computed by means of stochastic gradient descent (SGD) algorithm. The SGD is an optimization algorithm for minimizing the cost function. The algorithm takes steps proportional to the negative of the gradient.

Equation 2.5 defines the chain rule to calculate the derivatives to update the weight matrices ($\mathbf{W}$).

$$\mathbf{W}_{t+1} = \mathbf{W}_t - \eta \frac{\partial C}{\partial \mathbf{W}_t} \tag{2.5}$$

The gradient of the cost (C) is calculated with respect to each of the network weights ($\mathbf{W}$). The learning rate (η) is a value to determine how quickly the weights are adjusted. A good learning rate value is: (1) low enough to allow the network converges to generate good outputs, and (2) high enough to avoid spending too much time for training.

SGD can be slow when the local minimum is near. **Momentum** technique accelerates the training towards where SGD is heading. It works by having knowledge from the previous update values v_t which is a velocity vector stored for all the parameters. v_t is weighted by a new hyperparameter μ (see Equation 2.6) .

$$\mathbf{W}_{t+1} = \mathbf{W}_t - \mu v_t - \eta \frac{\partial C}{\partial \mathbf{W}_t} \tag{2.6}$$

Adagrad (Duchi et al., 2011) is an algorithm for gradient descent optimization that adapts the learning rate to the parameters according to the history of gradients. It performs larger updates for infrequent parameters and smaller updates for frequent parameters. Therefore, it is good to deal with sparse data. *Adagrad* uses a different learning rate η for every parameter at every timestep. We then set G_t which is a diagonal matrix containing the sum of the squares of the past gradients (see Equation 2.7).

$$\mathbf{W}_{t+1} = \mathbf{W}_t - \frac{\eta}{\sqrt{G_t + \varepsilon}} \frac{\partial C}{\partial \mathbf{W}_t} \tag{2.7}$$

where ε is a smoothing term that avoids division by zero. *Adagrad* basically divides the current gradient by the sum of previous gradients in each update. As a result, when the gradient is very large, η is reduced and when the gradient is small, η is increased.

The main advantage of *Adagrad* is that there is no need of tuning the learning rate because it is adapted automatically. By contrast, the accumulation of gradients grows during training and the learning rate can become very small with the risk that the algorithm cannot learn anymore.

Adadelta (Zeiler, 2012) is an extension of *Adagrad* which instead of accumulating all the past gradients, it restricts the number of gradients to a fixed size.

Instead of just storing the previous gradients, the previous gradients are recursively defined as a decaying average of all past gradients. Then, we substitute G_t by the decaying average over past squared gradients $E[g^2]_t$ (see Equation 2.8).

$$\mathbf{W}_{t+1} = \mathbf{W}_t - \frac{\eta}{\sqrt{E[g^2]_t + \varepsilon}} \frac{\partial C}{\partial \mathbf{W}_t} \tag{2.8}$$

where the current gradient is divided by the sum of previous gradients g and an offset ε in each update.

Adam (Kingma and Ba, 2014) also includes the methods incorporated in *Adadelta* and in addition it keeps an exponentially decaying average of past gradients, similar to momentum. The following formulation presents the first and second moment, respectively.

$$m_t = \mu_1 m_t + (1 - \mu_1) \frac{\partial C}{\partial \mathbf{W}_t} \tag{2.9}$$

$$v_t = \mu_2 v_t + (1 - \mu_2) (\frac{\partial C}{\partial \mathbf{W}_t})^2 \tag{2.10}$$

where m_t is the estimation of the first moment (the mean) of the gradients and v_t is the estimation of the second moment (the uncentered variance) of the gradients.

The authors realized that m_t and v_t are biased towards zero during the initial steps, so they computed a bias correction (see equations 2.11 and 2.12).

$$\hat{m}_t = \frac{m_t}{1 - \mu_1^t} \tag{2.11}$$

$$\hat{v}_t = \frac{v_t}{1 - \mu_2^t} \tag{2.12}$$

Then, the parameters are updated as described in Equation 2.13.

$$\mathbf{W}_{t+1} = \mathbf{W}_t - \frac{\eta}{\sqrt{\hat{v}_t} + \varepsilon}\hat{m}_t \tag{2.13}$$

According to the authors, *Adam* algorithm works better than other adaptive learning rate method algorithms.

For further information about SGD optimization algorithms, we refer the reader to Ruder (2016).

2.1.2 Solutions for vanishing and exploding gradient

Besides all the advantages of RNN, there exist two problems when dealing with very long sequences: the vanishing and exploding gradients. The gradient can be vanishingly small preventing the weight from changing its value. This can cause completely stop the neural network from training. As an opposite case, the gradient can be large. This can result in large updates to the network weights provoking an unstable network and being unable to learn from training data. At an extreme case, the weights values can become so large as to overflow and result in *NaN* values.

A solution for the vanishing gradient problem is the long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997). The **LSTM** incorporates gates to prevent the backpropagation of errors from vanishing flowing smoothly through time. The gates control how much an RNN wants to reuse memory (forget gates), receive input signal (input gates), and extract information (output gates) at each timestep.

Another type of RNN is the Gated Recurrent Units (GRU) (Cho et al., 2014b). The **GRU** is simpler to compute and implement than LSTM, it incorporates only two types of gates instead of the three types in LSTM. The hidden state is controlled by sophisticated units in order to memorize information or delete it (see Equation 2.14). The GRU contains reset and update gates. The reset gate (equivalent to the forget gate) allows the network to drop information that is irrelevant in the future (see Equation 2.15). On the other hand, the update gate (equivalent to the input gate) controls how much information from the previous hidden state will carry to the current hidden state (see Equation 2.16).

$$\mathbf{s}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \ \tanh\left(\mathbf{W}_s \mathbf{v}_t + \mathbf{r}_t \odot (\mathbf{U}_s \mathbf{s}_{t-1}) + \mathbf{b}_s\right), \tag{2.14}$$

$$\mathbf{r}_t = \sigma\left(\mathbf{W}_r \mathbf{v}_t + \mathbf{U}_r \mathbf{s}_{t-1} + \mathbf{b}_r\right), \tag{2.15}$$

$$\mathbf{z}_t = \sigma\left(\mathbf{W}_z \mathbf{v}_t + \mathbf{U}_z \mathbf{s}_{t-1} + \mathbf{b}_z\right), \tag{2.16}$$

where $\mathbf{v}_t$ is the input vector and $\mathbf{s}_t$ is the output vector at the timestep t. $\mathbf{r}_t$ and $\mathbf{z}_t$ are the reset and update gate activations, respectively. $\mathbf{W}_s$, $\mathbf{U}_s$, $\mathbf{W}_r$, $\mathbf{U}_r$, $\mathbf{W}_z$, $\mathbf{U}_z$ are the trained parameters and $\mathbf{b}_s$, $\mathbf{b}_r$ and $\mathbf{b}_z$ are the biases. *tanh* and σ refer to the hyperbolic tangent and the logistic sigmoid activation functions, respectively.

Cho et al. (2014b) included a GRU and Sutskever et al. (2014) used LSTMs to implement the first RNN based NMT systems.

Gradient clipping is a technique to avoid exploding gradient (Pascanu et al., 2012). Gradient clipping limits the magnitude of the gradient. The idea consists of checking if the norm of the final gradient g of the batch is greater than a threshold Γ. If it is the case, the scaled gradient $\frac{\Gamma}{||g||}g$ is used instead of g. This limits the gradient and retains the direction of training to avoid exploding of the gradient. Using gradient clipping the learning rate can be up so that the training converges much faster. The default option is to limit the norm of the gradient to be no more than the value 1.

2.1.3 Network initialization

Weight initialization of neural networks is an area of active research, it can help to obtain a better performance. Various methods have been proposed to deal with that. Uniform random initialization inside an interval is one of the first systematic method.

The **Xavier** (Glorot and Bengio, 2010a) algorithm automatically determines the scale of initialization based on the number of input and output neurons. The *Xavier* initialization formula is presented in Equation 2.17.

$$Var(\mathbf{W}) = \frac{2}{n_{in} + n_{out}} \tag{2.17}$$

where n_{in} specifies the number of input neurons, n_{out} is the number of output neurons and $Var(\mathbf{W})$ is the variance of the weights.

If the weights start too small, the information shrinks as it passes through each layer until it is too tiny to be useful. In the other extreme case, if the weights start too large, the information grows as it passes through each layer until it is too massive to be useful. Using *Xavier* initialization ensures that the learning of the weights starts correctly, keeping the information in a reasonable range of values through many layers.

2.1.4 Regularization

Deep neural networks usually contain a big number of parameters to successfully learn a function. However, large networks can overfit the model when adjusting too much the parameters to the given samples and not being able to generalize with new samples. Moreover, a large number of parameters makes the process of training and testing slow and difficult to deal with.

Regularizers are introduced in neural networks to prevent overfitting. One first thing to set is an appropriate number of hidden units in the model to learn the task with the given training dataset. Increasing the training dataset can help as well to avoid overfitting.

Weight decay

The simplest regularizer is weight decay. There are two types of penalty terms, L1 and L2. The L1 penalty term is proportionnal to the absolute value of the term. The L1 penalty term is applied to the cost function (see Equation 2.18).

$$\lambda \sum_t |\mathbf{W}_t| \tag{2.18}$$

where λ refers to the regularization parameter.

L1 regularization causes the weight to decay in proportion to its size.

The L1 penalty term is proportional to the absolute value of the term. The L1 penalty term is applied to the cost function (see Equation 2.18).

L2 regularization is proportional to the squared weights (see Equation 2.19).

$$\frac{\lambda}{2} \sum_t \mathbf{W}_t^2 \tag{2.19}$$

L2 regularization prevents the updates to be too large.

Equation 2.20 shows L2 regularization applied to the update rule.

$$\mathbf{W}_{t+1} = \mathbf{W}_t - \eta \frac{\partial C}{\partial \mathbf{W}_t} - \eta \lambda \mathbf{W}_t \tag{2.20}$$

Generally, in order to obtain a good performance using this regularizer, we can take into account: (1) the more training examples the weaker this term should be and (2) the more parameters the higher this term should be.

Performing such regularization helps with generalization, especially when training on small datasets.

Dropout

Dropout (Srivastava et al., 2014) is another regularizer technique to address the overfitting problem by modifying the network.

It works by randomly dropping units (along with their connections) from the neural network during training. This means that their contribution to the activation of downstream neurons is temporally removed on the forward pass and any weight updates are not applied to these connections on the backward pass. At test time, the network has all the outgoing weights halved.

If neurons are randomly dropped out, other neurons will have to step in and handle the representation required to make predictions for the missing neurons. This is believed to result in multiple independent internal representations being learned by the network. The effect is that the network becomes less sensitive to

the specific weights of neurons. This prevents units to overfit and the network can generalize better.

2.2 Neural Machine Translation

All the systems described in this document are based on the NMT system provided with attention mechanism (Bahdanau et al., 2014). It consists of a sequence to sequence encoder-decoder model with attention.

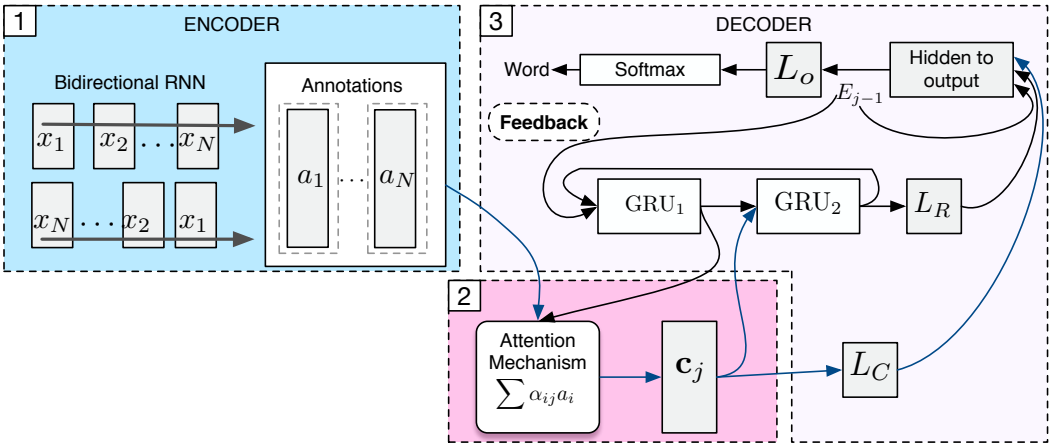

Figure 2.2: Attention-based NMT system.

2.2.1 Encoder

The encoder is a bidirectional RNN (see box number 1 of Figure 2.2) equipped with an attention mechanism. Each input sentence token x_i ($i \in 1 \dots N$ with N the source sequence length) is encoded into an annotation a_i by concatenating the hidden states of a forward and a backward RNN ($\overrightarrow{h_j}, \overleftarrow{h_j}$) with GRU (see Section 2.1.2). The hidden states are defined as follows.

$$\overrightarrow{h_i} = f(\overrightarrow{h_{i-1}}, \mathbf{E}x_i) \tag{2.21}$$

$$\overleftarrow{h_i} = f(\overleftarrow{h_{i+1}}, \mathbf{E}x_i) \tag{2.22}$$

where $\mathbf{E}$ represents the embedding lookup for each input word x_i

Each annotation in $\mathbf{a} = a_1 \dots a_N$ is a representation of the whole sentence with a focus on the current token.

$$a_i = \begin{bmatrix} \overrightarrow{h_i} \\ \overleftarrow{h_i} \end{bmatrix} \tag{2.23}$$

2.2.2 Decoder

The decoder contains a conditional gated recurrent unit (cGRU) (Firat and Cho, 2016) (see Section 2.1.2) consisting of two GRUs interspersed with the attention mechanism (see box number 3 of the Figure 2.2).

The first GRU (GRU_1 in Figure 2.2) cell of the decoder is fed by its previous hidden state and the feedback (*i.e.* the previous generated symbol) with the following formulation.

$$\mathbf{s}'_j = \mathrm{GRU}_1\left(\mathbf{E}[y_{j-1}], \mathbf{s}_{j-1}\right) \tag{2.24}$$

where $\mathbf{E}[y_{j-1}]$ is the embedding of the previous output y_{j-1} used for feedback.

GRU_1 is initialized at timestep 0 with the mean of the annotation vector $\mathbf{c}_j$ (see Equation 2.25).

$$\mathbf{s}'_0 = \tanh\left(W_{init}\left(\frac{1}{N}\sum \mathbf{a}_i\right) + \mathbf{b}_{init}\right) \tag{2.25}$$

The second GRU (GRU_2 in Figure 2.2) is fed by $\mathbf{s}'_j$ (the output of GRU_1) and the context vector $\mathbf{c}_j$, with the following formulation.

$$\mathbf{s}_j = \mathrm{GRU}_2\left(\mathbf{s}'_j, \mathbf{c}_j\right) \tag{2.26}$$

The initialization of the GRU_2 is done with the first timestep of GRU_1.

$$\mathbf{s}_0 = \mathbf{s}'_1$$

The output layer L_O is connected to the network through a sum operation inside of an hyperbolic tangent function $\Phi(\sum)$ which takes as input and sums the embedding of the previous generated token as well as the context vector and the output of the decoder from GRU_2 (both adapted with a linear transformation, respectively, L_C and L_R). Then, there is another layer L_O specialized on the output. Finally, the output probabilities for each token in the target vocabulary are computed with a *softmax* function. The token with the highest probability is the translation output at each timestep.

2.2.3 Attention mechanism

The attention mechanism (see box number 2 of the Figure 2.2) computes a source context vector $\mathbf{c}_j$ as a convex combination of annotation vectors, where the weights of each annotation are computed locally using a feed-forward network. The context vector $\mathbf{c}_j$ is defined in Equation 2.27.

$$\mathbf{c}_j = \sum_{i=1}^{N} \alpha_{ij}\mathbf{a}_i \tag{2.27}$$

where $\mathbf{c}_j$ is the sum of the annotations multiplied by the corresponding weights, being j the iteration of the output words.

α_{ij} weights have been normalized across all input words i using a softmax.

$$\alpha_{ij} = \frac{e^{e_{ij}}}{\sum_i e^{e_{ij}}}$$

These weights can be interpreted as the alignment score between target and source tokens. For each generated token at the target side, the model finds the relevant source context.

The feed-forward network for the attentional weights has been computed in Equation 2.28.

$$e_{ij} = \mathbf{v_a} \tanh(\mathbf{W}_D \mathbf{h}'_j + \mathbf{W}_E \mathbf{a}_i) \tag{2.28}$$

where $\mathbf{W}_D$, $\mathbf{W}_E$ represent the weight matrices transformations from the decoder and encoder, respectively and $\mathbf{a}$ corresponds to a weight vector.

First NMT models compute the probability of the target sentences conditioned on a fixed-length vector representing the whole source sentence. This fixed size representation degrades when sentence length increases.

Instead of encoding the input sequence into a single fixed context vector, attention mechanism allows the decoder to attend to different parts of the source sentence at each decoding step. Therewith the model learns how to attend based on the input sentence and what it has produced so far in order to generate a context vector for current decoding step.

2.2.4 Beam search

By applying a beam search to find the best translation or n-best translations, we can predict n-best output words at each timestep.

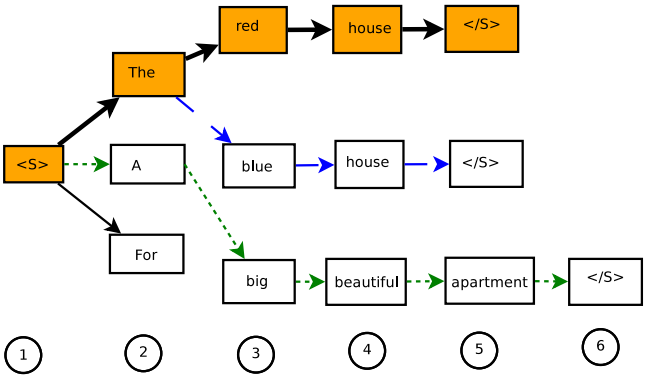

Figure 2.3: Example of beam search of size 3 generating sentence hypothesis in 6 timesteps.

See Figure 2.3 for an illustration (example with beam size set to 3). The beam search starts with the beginning of sentence symbol `<s>` (timestep 1). When predict-

ing the first word of the output sentence, the words are scored by their probability and the beam size hypothesis with the highest scores are kept (timestep 2). Then, we use each of these words in the beam as conditioning context to generate the next word predictions (timestep 3). Afterwards, we multiply the score of the partial sentence hypothesis with the probabilities of the word predictions. The number of sentence hypothesis with the highest accumulated score equal to the beam size are kept (timesteps 3, 4, 5 and 6). This procedure continues predicting next words until an end of sentence symbol `</s>` is predicted with the highest probability (timesteps 5 and 6). At this point, the completed hypothesis are kept as n-best translations and the beam size is reduced by the number of finished sentence hypothesis. When all the sentence hypotheses are finished, the search process ends (timestep 6). Finally, the word prediction probabilities of each sentence hypothesis are multiplied and the sentence hypothesis with the highest score is selected as the best translation. Normally, we normalize the score by the output length of a translation to get better results.

In the example given in Figure 2.3, three hypothesis have been obtained. The best translation is shown in background color and with the thickest arrows pointing the path through all the words in the sentence. Two more sentence hypothesis have been generated as 2nd (blue and thick discontinuous arrows) and 3rd (green and thin discontinuous arrows) best translations.

The probability distribution in neural machine translation is often spiked having few hypothesis. In statistical machine translation, the hypothesis can be combined if they share the same conditioning context for future predictions. This is not possible for recurrent neural networks since the entire output word sequence is conditioned from the beginning. As a consequence, the search graph is generally less diverse than search graphs in statistical machine translation. Therefore, there are less different hypothesis and beam search or rescoring practices do not obtain that much improvement.

2.2.5 Evaluation and early stopping

NMT model is trained with the backpropagation algorithm using training dataset. After some updates, the model is evaluated using the development dataset. This step is called validation. The automatic metric chosen to evaluate the model is Bilingual Evaluation Understudy (**BLEU**) (Papineni et al., 2002a). It is a score which takes values between 0% and 100%. This metric is currently one of the most popular in the field.

BLEU is an algorithm for evaluating the quality of text which has been machine-translated from one natural language to another. Quality is considered to be the correspondence between a machine's output and that of a human: the closer a machine translation is to a professional human translation, the better it is.

BLEU measures the n-grams precision with respect to a set of reference translations, with a penalty for too short sentences. In practise, BLEU implements a geometrical average of n-gram ($n = 1$ to 4) precisions. The consequence of this is

that BLEU is well-defined at corpus level, and not at sentence level.

BLEU can be single or multi-reference. If several references are provided in the dataset, BLEU has a fairer score for an automatic translation. This is because one source sentence can have several correct translations. A higher BLEU score does not mean in all cases an actual improvement in translation quality. There are works showing some examples where BLEU does not correlate with human judgments (Callison-Burch et al., 2006).

The validation step is repeated every some updates after typically complete the first epoch. In every validation we keep the top n-best models. After some validations without improvement of BLEU score, the training stops, this process is called **early stopping**.

In this thesis, attention based NMT system is used to train all the models.

CHAPTER 3

STATE OF THE ART

Contents

> "Language is a complex beast. At best, machine translation is a reasonable alternative to zero translation". Don DePalma

Despite all the recent success developing neural machine translation (NMT) systems, there are still remaining challenges to improve the translation performance.

For example, translating rare words is a difficult task, one of the reasons is because the vocabulary size is limited due to the complexity of the softmax function. On the other hand, classical phrase-based systems are not that sensitive to rare words because they can handle large vocabulary in their phrase tables and their very defined alignments. We will present in this chapter the state of the art approaches to handle larger vocabularies in neural networks. We will continue by explaining the usage of factored models developed in statistical machine translation, neural language modeling and neural machine translation to mainly overcome the limitation of the vocabulary size. Finally, we will end up introducing some models that handle multiple inputs or outputs for different tasks and modalities.

3.1 Dealing with large vocabularies

Several proposals have been put forward to address the problem of large vocabulary management.

3.1.1 Shortlist

The classical approach to handle the large vocabulary with the softmax layer size limitation is called *shortlist*. It consists of selecting only the most frequent words to create the vocabulary and the rest of the words are mapped to unknown word (*i.e.* `UNK` token). The softmax operation is calculated over the shortlist to extract the highest probabilities. This approach is simple but it models only a limited part of the vocabulary, generating many unknown tokens.

3.1.2 Unknown words replacement

Luong et al. (2015b) addressed the problem of rare words in NMT making use of their alignments. The unknown generated words are substituted in a post-process step by the translation of their corresponding aligned source words or copying the source words if no translation is found. The model used in this work is the model of Sutskever et al. (2014), one of the first ones, which does not contain attention mechanism. Therefore, they use an unsupervised aligner to produce the alignments. The translation of the corresponding source words is made using a bilingual dictionary. Their unknown words replacement technique improves 2 BLEU points the system. Afterwards, Jean et al. (2015b) made use of the attention mechanism of their model to extract the alignments. Introducing this technique, they obtained competitive results compared to SMT systems ranking the first in the Workshop of Machine Translation (WMT) 2015. The drawback of this method is the use of a postprocessing step that it is not integrated into the training process of the model. Moreover, the obtained word based translation is not using the context. Therefore, in many cases, it can return a wrong translation when the word is ambiguous.

3.1.3 Synthetic data

Recently, in SMT, the unseen morphological variants of words are added as synthetic phrases for decoding (Huck et al., 2017b). Then, the system incorporates the information of all the inflections of the seen words in the training set.

Another attempt in SMT is the back translation technique (Bojar and Tamchyna, 2011). Training a model in the opposite direction to translate lemmas makes possible to create synthetic parallel data. It might contain unseen word forms of known lemmas on target side. The drawbacks are that the source side is automatic translated and it can have some errors. Also, the substitution of words with lemmas may loose some translation information.

The monolingual backtranslated data has been also incorporated into NMT (Sennrich et al., 2016b) at word level. Monolingual data is automatic translated with a model trained in the opposite direction creating the synthetic parallel data. This allows the system to manage more quantity of training data boosting the translation performance.

3.1.4 Structured output layer

One strategy used in neural language modeling (NLM) is to define a structured output layer (SOUL)(Le et al., 2011) to handle the words not appearing in the shortlist. Several softmax layers replace the output layer. The output consists of a tree structure with a main softmax and a node which contains sub-class layers with their own softmax. The main softmax contains the most frequent words. The rest of the words are separated in classes and for each class a smaller softmax is used. This allows the system to always apply the softmax normalization on a layer with reduced size. The structured output layer NLM was applied for speech recognition and some benefits were shown with languages with large vocabularies like highly inflected ones. Another similar works are about hierarchical language models where the words are also organized in clusters in a tree structure (Mnih and Hinton, 2008; Chen et al., 2015).

The pros of this strategy is that the model can manage more vocabulary, facing the problem of the limited size of the softmax and reducing the number of the unknown words. On the contrary, the structure output layer has the disadvantage of dealing with a complex architecture. It requires to prepocess the vocabulary in classes and training with much deeper architecture.

3.1.5 Sort batches by subsets of vocabulary

Another work around for NMT has been proposed in Jean et al. (2015a). They 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. On the contrary, this technique faces the problem of mismatching the

way of training and testing the model because at test mode the batches are not sorted.

In a similar way, Chen et al. (2015) used the same technique in order to train large vocabulary for neural language modeling. Additionally, the same work shows a similar technique which distinguishes between genuine data and noisy samples to sort the batches in a better way.

3.1.6 Subwords

An alternative approach is to work with representations designed to remove some variations via source-side or target-side splitting procedures.

The most successful proposal is using subwords (Sennrich et al., 2016c) which seems to achieve a right balance between a limited vocabulary size and the ability to translate a fully open vocabulary. In a nutshell, this approach decomposes source and target tokens into smaller units of variable length (using what is now termed as a "Byte Pair Encoding" or BPE in short): this means that (a) all source tokens can be represented as a sequence of such units, which crucially are all seen in training; (b) all possible target words can also be generated; (c) the size of the output layer can be set to remain within tractable limits; (d) most frequent words and subwords are kept as BPE units, which preserve the locality of many dependencies. The vocabulary can be shared for both languages (joint or bilingual vocabulary) helping to generate, for example, proper names that are already in the source language and they do not change in the target language.

The BPE algorithm has as unique parameter the number of merge operations and consists of several steps:

1. Read the training and validation datasets at character level.
2. Merge the most frequent tokens and repeat merge process, according to the number of merge operations given as parameter (as shown in the line below).

   a a a b c a a a b a →*aa a b c aa a b a* →*aa ab c aa ab a* →*aaab c aaab a*
3. With the resulting text, build a BPE tokens dictionary specifying word endings.
4. As preprocessing of the datasets, split words according to the BPE dictionary adding a special suffix (@@) to specify the subwords.
5. Once the preprocess is done, training and test can be performed.
6. As postprocessing of the generated output, the subwords have to be reconverted into words (as the example below).

   Ex@@ ample of sub@@ words →*Example of subwords*

This method is simple and it does not require external resources. The alignments can benefit of those splitting matching main parts of the word and prefixes or suffixes

of both source and target languages. Furthermore, it can generate words unseen in the training data, it only needs that the subword appears in the corpus. Also, it drastically reduces the generation of unknown words. On the other hand, BPE can generate incorrect when combining different subwords. BPE is the main method used in recent machine translation evaluation campaigns (WMT, IWSLT).

3.1.7 Character-level Neural Machine Translation

More extremely representation of words is considering translation at character-level unit. In the character-based NMT proposed by Ling et al. (2015), words are modelled at character-level. As input of the network, they use word vectors represented by characters. At the output side, the model generates sequences of word vectors where each word is generated one character at a time. The authors say that the method can improve over a equivalent word-level model.

Costa-jussà and Fonollosa (2016) still uses word embeddings but instead of being an independent vector, they are computed from the characters embeddings of the corresponding word. However, they still have limited word target vocabulary.

For the first time, Chung et al. (2016) answered the question of whether NMT can be done directly on a sequence of characters without any word segmentation. They implemented a model using subwords in the input and characters in the output of the network. Improvements in the translation performance compared to fully BPE level models are reported.

A hybrid system translating mostly at the word-level and character-level for the rare words was implemented by Luong and Manning (2016). It is easier and faster to train than fully character-level and it never generates unknown words but it cannot benefit of common lexemas between words most of the times.

Another similar method where words and character are mixed has been done (Wu et al., 2016) arguing that they find a good balance of vocabulary using both segmentations.

Subsequently, fully character-level NMT without explicit segmentation was implemented (Lee et al., 2016). This method maps a source character sequence to a target character sequence without specifying the word segmentation. This implementation is made using convolutional networks with max-pooling at the encoder to reduce the length of character source representation. In this way, the training speed will not be increased. They applied the method also in a many-to-one translation task making profit of the common morphemes between languages and without increasing the model size. They also argue that without using word boundaries, the model can learn an internal structure of a sentence to map the symbols in a continuous representation. They report some improvement when comparing with BPE units translation.

The advantages of character-level NMT is that all the vocabulary can be covered with a small size of softmax layer, it can model morphological variants of a word and it avoids problems in preprocessing/tozenization. Moreover, unseen words can be generated as with BPE. However, character-level NMT is much more costly to

train to have a good performance than higher level representation and it makes long distance dependencies even longer.

The same author who applied the BPE approach (Sennrich, 2016) showed a comparative study about the character-level translation quality utilizing a special test set of contrastive translation pairs. He reports that character-level NMT systems perform better at transliteration and processing unknown words than models with BPE segmentation. By contrast, character-level systems perform worse than BPE units systems when extracting morphosyntactic agreement and translating discontinuous units of meaning.

3.2 Factored models

Several linguistic resources have been used in machine translation in order to improve the MT quality. They can be used to increase the supported vocabulary, creating new unseen word forms or adding more context to the translation. In this section, we explain the state of the art of using factors in statistical machine translation, language modeling and neural machine translation.

3.2.1 Factors in Statistical Machine Translation

Factored models were first invented by Bilmes and Kirchhoff (2003) where factors are used as additional information for language modeling. Afterwards, factored translation models were introduced for statistical machine translation (SMT) (Koehn and Hoang, 2007). In this work, the authors incorporate additional word level features at source and target side of the model. These features consist of the linguistic information of the words like lemma, Part of Speech (POS) tag, morphological information and an automatically generated word class. Afterwards, they extended their work to support the translation of longer phrases composed of factors to act as templates for the selection and reordering of surface forms (Hoang and Koehn, 2009). The factored translation models (Koehn and Hoang, 2007) were also applied for Arabic translation (Youssef et al., 2009) showing a tendency to improve the morphology of the translation resulting in a better performance. Factored language models were also applied to SMT taking into account larger bilingual context (Crego and Yvon, 2010).

At the same time, other approaches utilize a set of syntactic and morphological (*stems* + factors) knowledge sources from both source and target sentences in a SMT model (Minkov et al., 2007; Toutanova et al., 2008) to translate into Russian and Arabic. The surface form of the words is selected from the full paradigms in a second step using a classifier. They show that the use of morphological and syntactic features leads to large gains in prediction of the accuracy. Another work used case markers in addition to words in order to improve the fluency in English to Hindi translation (Ramanathan et al., 2009).

Another method consists of improving SMT for morphologically rich languages using a hybrid morpheme-word representation (Luong et al., 2010). Target words

may also be represented as lemmas complemented with side information. Bojar (2007); Bojar and Kos (2010); Bojar et al. (2012) use such representation for two SMT systems: the first one translates from English into Czech lemmas with source-side information and the second one performs a monotone translation into fully inflected Czech. Jawaid and Bojar (2014) use in the first step a hierarchical system that outputs a lattice presenting different word orders. Then, the second step selects the word order that allows the system to choose the best morphological predictions.

Haddow and Koehn (2012) introduced backoff methods by combining surface forms and factored forms of the words. This method showed gains in performance and improved the accuracy of the rare words translation.

Fraser et al. (2012) proposed a target morphology normalization for German words represented as lemmas followed by a sequence of morphological tags. They introduced a linguistically motivated selection of words when translating from a morphologically poor language as English to a morphologically richer language as German. The selection step consists of predicting the tags that have been removed during normalization, using a specific Conditional Random Field (CRF) model for each morphological attribute to predict. Finally, word forms are produced via look-up in a morphological dictionary. This approach is extended by Weller et al. (2013), who takes verbal subcategorization frames into account, thus enabling the CRFs to make better predictions. Related approaches have been proposed (Burlot et al., 2016) for Czech and Arabic (El Kholy and Habash, 2012b,a). Weller-Di Marco et al. (2016) handle the prediction of both prepositions and morphological features by building synthetic phrase tables.

Linguistics factors have been used as well for Hindi to rerank automatic translation n-best hypothesis (Gupta et al., 2014). In this system, they produced stemmed and POS tagged text to train a trigram language model. The results showed that the ranking performed as human judgement.

Some studies analysed the usefulness of factors in SMT (Durrani et al., 2014). When morphological information is not available for resource poor language, they cluster the words automatically. The authors saw an average improvement of +0.8 BLEU points using those factors.

3.2.2 Factors in Neural Language Model

Factors as extra information of the words have been utilized in some works to provide linguistic grammatical information to the neural networks. Neural language models have been frequently used to add features to the SMT systems (Schwenk, 2010) giving more context information for long sentences. Several works have used factors as additional information for the input words in neural language modeling (NLM) with good results (Niehues et al., 2016; Alexandrescu, 2006; Wu et al., 2012).

Niehues et al. (2016) used a factored word representation at output side as well as input side for NLM. Their RNN based network consists of multi-factors input and output sides that facilitates all available information about the word. They make use of the surface form, POS-tag and automatic word clusters of 100 and 1000

sizes based on Och (1999). For each factor, an embedding is learnt by the network. Then, the concatenation of the embeddings is used to represent the word. In the last layer, one softmax is used for each output factor. For the target side, also the source word is kept as additional factor. The NLM is used as additional feature for rescoring an nbest list in a phrase-based MT system.

Baltescu et al. (2014) present an open source implementation of an NLM toolkit. This toolkit provides new factored features based on Brown clustering (Brown et al., 1992) to improve the SMT quality. It facilitates the NLM integration as a feature in the beam search of the hierarchical phrase-based translation system *cdec* and *Moses* decoders.

3.2.3 Factors in Neural Machine Translation

Recently, factors have also been integrated into a word-level NMT system as additional linguistic input features (Sennrich and Haddow, 2016). The concatenation of all the features embeddings is learnt by the encoder.

Syntatic information in the form of Combinatorial Categorial Grammar (CCG) supertags was integrated for NMT task (Nadejde et al., 2017). For the source side, they used the same approach as Sennrich and Haddow (2016). Three different strategies were experimented for the target side. The first one is called serializing where the target supertags together with the subwords are interleaved in the same target sequence increasing the sentence length. The other two strategies are about multitasking, one, sharing the encoder but using different decoders for each output (CCG tag and translation) with the disadvantage of having no way to restrict the number of predicted words and tags to be able to match them. The other multitasking strategy consists of sharing encoder and decoder excepting the softmax to predict each output. They do not connect the generated CCG supertag to the predicted translation due to the required modification of the beam search. They showed improvements in the translation performance with the serialization strategy.

Eriguchi et al. (2016) integrate source-side syntax of the phrases alignments between English and Japanese. They build a tree-based encoder following the parsed tree allowing the attention mechanism to align not only words but also phrases with the output words.

More recently, a neural machine translation system was trained to predict interleaved lemma and morphologically rich POS tag in an output sequence duplicating the length of the sentence (Tamchyna et al., 2017). In a second step translation, the surface form of each word is generated from its predicted lemma and POS tag. The model consists of an encoder-decoder NMT system with attention mechanism and using BPE method. The strategy is applied for English→Czech and English→German language pairs translation. They argue that the presence of lemmas allows the system to model inflections and capture lexical correspondence with the source. Unseen words also can be generated but their improved results are not mainly because of this reason. They found that the benefit comes from the words decomposition. They also predict the surface and POS tags, similarly as Nadejde

et al. (2017) did with CCG supertags but they could not improve the results compared to the baseline. They attribute their results to the trade-off between providing explicit morpho-syntactic information (weaker than CCG supertags) and increasing the sequence length.

Furthermore, recent works show alternatives of using BPE segmentation in NMT. For Turkish, Ataman et al. (2017) used a supervised and unsupervised morphological segmentation. This method splits the words preserving their morphological information, which BPE sometimes breaks. As well as BPE method does, they proposed segmentation also reducing the vocabulary size to fit a given constraint. The results of this approach showed better performance than BPE method due consideration of linguistic notion in the segmentation process. This unsupervised method to extract factors is specially useful if no linguistic resources are available for a given language. Huck et al. (2017a) present an approach for German segmentation using compound splitting based on the stems. They conclude that linguistically motivated target-side word segmentation improves NMT when translating into an inflected and compounding language. Another approach is the one designed by Sánchez-Cartagena and Toral (2016) where they use a tool to extract the morphs instead of using the words to translate Finish. They apply on top of the morphs the BPE method increasing the length of the sentences. This method was applied for the WMT evaluation campaign and ranked in the first group.

There are many differences between SMT and NMT, one of them is that NMT handles larger context than SMT. This makes NMT more capable to capture the word morphology in the layers of the network. The quality of the representations in the NMT model for learning sentence morphology has been evaluated in Belinkov et al. (2017). They analyse different word representations (character-based and word-based) concluding that character-based ones are better for morphology learning, especially with rare unseen words. On the other side, word-based models are enough to learn the structure of frequent words. They studied different depths of neural network showing that lower layers in the encoder are better to capture the morphology, in contrast to deeper layers which are more focused on word semantic improving the translation quality. This study was performed on different language pairs and they argue that translating into morphologically-poor languages lead to better source-side word representation than target-side. They also inform that the decoder does not learn well the morphology, mainly because the attention mechanism focuses on the relevant parts of the source word representations.

3.3 Multiple inputs and outputs networks

There have been other attempts with dual training objectives and several input information in NMT.

3.3.1 Multi-task Neural Networks

A linguistic multitask approach has been developed (Niehues and Cho, 2017) to generate with the same model the translation, named entities and POS-tags of a sequence. By jointly training several natural language processing (NLP) tasks in one model, it is possible to get benefit of common information in order to improve machine translation performance. The training is done using separated minibatches per each task. The corpus used is small and different for each task as well. The authors experimented different options of shared parameters between the tasks: (1) sharing only the encoder, (2) sharing the encoder and the attention mechanism and (3) sharing all except the softmax layer. The first option sharing less parameters obtained better result. They showed an improvement in the performance of the translation and the POS tagger using the multitask model under low-resource conditions.

In Chen et al. (2016), a *guided alignment training* using topic information of the sentence as a second objective helps the decoder to improve the translation.

Multi-task cross-lingual sequence models have been developed for tagging purposes (Yang et al., 2016). They built two kinds of models. The first one is a multi-task model where multiple objectives (Part-Of-Speech tagging, chunking and named entity recognition) are predicted in the output layers of the network. The architecture consists of separated layers for each specific task and shared layers for the rest of the parameters (encoder and GRU). The second model, also incorporates multilingual learning for named entity recognition task. The architecture of this second model shares the weights of a character-based GRU and the rest of the architecture is specific for each language including word embeddings and word-based GRUs. They could improve performance in several cases when using this joint training.

3.3.2 Multilingual Neural Machine Translation

Multi-task learning for multilingual purpose has been investigated. First, Dong et al. (2015) built a model where multiple output languages are translated sharing the same source language encoder. Each language pair has its own attention mechanism. Also, basic encoder-decoder networks with single-source attention have been utilized for multiple input languages to translate them into one target language (Zoph and Knight, 2016).

Multiple input and output neural networks have been developed recently (Firat et al., 2017) using scheduled decoders with multiple source and target languages. This system was designed to interact with data from multiple language pairs creating a space where common multilingual information is kept in the attention mechanism. They experimented with five different languages in the source, each one has an encoder and five languages in the target size which each one has their own decoder. All the languages share the same attention mechanism and transformers are used to match the different dimensions of each language. They schedule the training for

each different language pair, updating only the parameters of a language pair per batch.

A multi-task strategy has been used for multilingual task purpose (Ngoc-Quan et al., 2017). They propose several settings: (1) sharing all parameters of the encoders, decoders and attention for all the language pairs, (2) encoders and decoders are shared but attention is separated for each language pair and (3) encoders and decoders are language specific and attention is separated. The results showed that the second option sharing encoders and decoders performed better. They also conducted zero-shot translation and having the language as a word feature improved the scores. Zero-shot translation has been also tried by Surafel et al. (2017). Using pivoting techniques, they showed the potential and benefits of using a multilingual setting. Results were better for multilingual model when comparing it with single language pairs models under low resource conditions.

3.3.3 Multimodal Neural Machine Translation

Other works using multiple output networks are focused on multimodal tasks. Multilingual image description task systems have been implemented with the NMT encoder-decoder approach to capture a representation of the image and the source language words to produce descriptions in the target language (Elliott et al., 2015).

Multi-task models for translation and image captioning also integrated the sequence to sequence approach (Luong et al., 2015a). Other related works are able to produce neural image caption generation using convolutional neural networks to encode an image and recurrent neural networks to generate the captions. Finally, in other work, Vinyals et al. (2015) train the model to maximize the likelihood of the sentence given the image . Moreover, this same model has been extended with attention mechanism (Xu et al., 2015) showing the correlation with the learned alignments and the human intuition.

Part II

Contributions

CHAPTER 4

FACTORED NEURAL MACHINE TRANSLATION

Contents

"Translation is not a matter of words only: it is a matter of making intelligible a whole culture."

Despite all the advantages discovered in NMT systems, we have seen that there still exist some challenges to face in order to obtain better performance. In this chapter, we introduce a new approach which faces the NMT challenges like the limitation of the vocabulary size and the data sparsity. For that, we use factors based on linguistic features. We first describe the factored approach motivation and its architecture which incorporates several sequences on one side of the network. Then, we evaluate the factored approach in several scenarios. The first set of experiments basically compares factored approach with other state of the art approaches. The second set of experiments is about the design of some options of the architecture. Then, the last set of experiments presents a new evaluation with a new preprocessing of the data including a qualitative analysis of the translation outputs. Finally, the factored approach is applied at both sides of the network applying different types of attention mechanism. Moreover, the factored approach is combined with word-level and BPE-level at input and output sides comparing the translation performance.

4.1 Motivation

Factored Neural Machine Translation (FNMT) approach consists of replacing the plain word representation with its lemma and additional tags (factors). In morphology, a lemma is the dictionary form or headword of a set of words. For example, “are”, “were”, “was”, “being”, “is” are inflections of the same lemma, which is “be”. Lemmas have special significance in highly inflected languages such as Arabic, Turkish and Russian. By using the lemmas, we are able to generalize to bigger vocabulary. In this thesis, *factors* are referring to some linguistic annotations at word level, *e.g.* the Part of Speech (POS) tag, number, gender, etc. We include factors jointly with the lemma to indicate how to inflect the lemma to be able to transform it to the surface form. We focus on the inflection morphological phenomena. Other morphological phenomena as agglutination are not target. Linguistic factors applied to NMT have been introduced for the first time in this work.

In standard NMT, the tokens are not linked with all their morphological variations and there is no explicit information about morphological features. By contrast, the use of lemmas directly in the NMT models allows the system to connect all the inflections of the lemmas and capture lexical correspondence. In addition, factors may help the translation process providing grammatical information to enrich the output. We are able to simplify the learning and alignment of the related words that share the lemma or the factors. We argue that when parallel data is sparse, taking into account other characteristics of the data as linguistic factors can help to improve translation performance. FNMT method is specially useful for low and

medium resource setting to translate into highly inflected languages because of its generalization data properties.

Another problem in NMT is the generation of unseen surface forms in the training data. Knowing the lemmas and their factors, the FNMT system can generate all their inflections without explicitly seeing them in the training data. Moreover, having the PoS tag can be useful for disambiguation (*e.g. book*: noun or verb). Humans also do this for language generation knowing only the base word and how to inflect it. By integrating directly linguistic information into the NMT models, we expect to improve the translation performance.

Subword segments produced by the BPE method are able to generate unseen new words. However, since they are not linguistically informed, they can produce erroneous surface forms by concatenating several subwords. Furthermore, words can be only produced by joining subwords from the dictionary produced.

4.2 Description of the approach

FNMT approach differs from preceding works in the sense that it uses only the linguistic decomposition of the words (no surface form) and predicts simultaneous outputs at the target side of the network. Additionally, it can be also applied at the source side.

Figure 4.1 presents the general architecture of the FNMT system where two different outputs are generated simultaneously: the lemma and its factors depending on the language. Indeed, each word is represented by its lemma and its linguistic factors (POS tag, tense, gender, number, person, etc). By these means, the target vocabulary size is reduced because all the derived forms of the verbs, nouns, adjectives, etc. are not kept. We can also apply factored level at input side of the network to get its advantages in the source language as well.

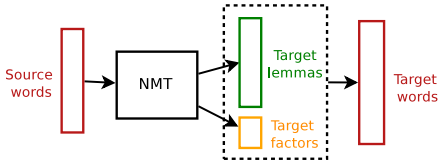

Figure 4.1: NMT system pipeline with factored output

For simplicity reasons, only two symbols are generated: the lemma and the concatenation of the different factors that are considered. We are able to reduce one big output vocabulary (surface forms) into two vocabularies: one for lemmas and a very small vocabulary for the factors (see Equation 4.1).

$$|V_{words}| > |V_{lemmas}| \gg |V_{factors}| \tag{4.1}$$

Given both output sequences (lemma and factors) and linguistic resources, the final surface form is easily generated with a minimum of ambiguity.

This approach handles the challenge of the limitation of the output layer size containing the target language vocabulary using factors as a translation unit. The low frequency words in the training set can benefit from sharing the same lemma with other high frequency words, and also from sharing the factors with other words. The vocabulary of the target language contains only lemmas and PoS tags but the total number of surface words that can be generated (i.e. virtual vocabulary) is larger (see Equation 4.2). This allows the system to correctly generate words which are considered as unknown in word-based NMT system.

$$|V_{words}| \ll |V_{lemmas}| \times |V_{factors}| \tag{4.2}$$

4.2.1 Preprocessing: from words to factors

We make use of a POS-tagger to extract the factors from the words. The lemma tokens are separated in a first sequence. The rest of the tokens (POS-tag, gender, number, etc.) are concatenated in a second sequence.

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 (see Table 4.1).

word	lemma	factors:	PoS tag	tense	person	gender	number
devient	devenir	VP3#S:	verb	present	3rd	#	singular

Table 4.1: Example of extraction of factors from a word.

Morphological analyser

In order to extract the lemmas and factors, a linguistic tool is necessary. In our first experiments, target language is French and the factors POS-tag, tense, number, person and gender have been chosen for this language. The morphological and grammatical analysis is performed with the MACAON toolkit (Nasr et al., 2011). MACAON POS-tagger outputs the lemma and factors for each word taking into account its context. The Lefff dataset (Sagot, 2010), a large-coverage morphological and syntactic lexicon for French, is used by MACAON to build the models.

4.2.2 Main architecture

The FNMT model allows the system to generate several output symbols simultaneously as presented in Figure 4.2.

The FNMT architecture is an extension of the standard NMT one. The encoder and attention mechanism remain without modifications with respect to the base model (see Chapter 2, Section 2.2). However, the decoder has been modified to produce multiple outputs. The synchronicity of the two outputs generation is possible because the hidden states are shared between the two of them. The hidden

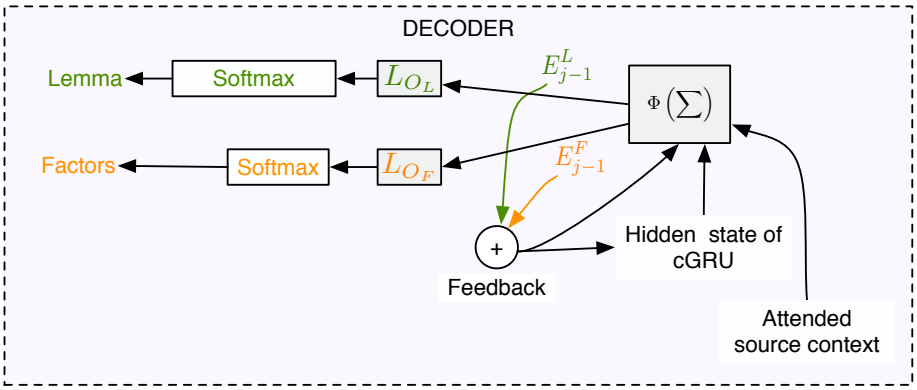

Figure 4.2: Detailed view of the FNMT system decoder.

to output layer ($\phi(\sum)$) is the sum of three inputs: (1) hidden state, (2) source context provided by the encoder and the attention mechanism and (3) feedback. In the NMT approach, the embedding of the previous word is given as feedback for generating the next word. For the FNMT approach, multiple outputs are available which can be possibly combined and given as feedback. In the first experiments (Sections 4.3 and 4.4.1), the feedback of the model contains only the embedding of the previously generated lemma which contains the main information of the surface form. Several options for feedback have been explored in Section 4.4.2. Finally, in the last part of the model, the output is split into two specialized output layers L_{OL} and L_{OF} which in turn feed a specialized softmax layer, one to calculate the lemmas and the other to calculate the factors.

An error or cost is calculated at each output which has to be backpropagated through the network. The cost from the first output containing lemmas is presented in Equation 4.3 and the one from the second output containing factors is shown in Equation 4.4.

$$C(\theta)^L = \sum_{(x,y^L)\in D} -\log p(y^L|x,\theta) \tag{4.3}$$

$$C(\theta)^F = \sum_{(x,y^F)\in D} -\log p(y^F|x,\theta) \tag{4.4}$$

where D refers to the sentence pairs being x the source sentences, y^L the lemma target sequences and y^F the factors target sequences.

The cost produced when training the model is obtained summing both costs, the one coming from the learning of lemmas output and the one coming from the learning of factors (see Equation 4.5)

$$C(\theta) = C(\theta)^L + C(\theta)^F \tag{4.5}$$

When applying this cost for backward pass (explained in Chapter 2, Section 2.1.1),

the shared weights between both outputs (from input to $\phi(\sum)$ in Figure 4.2) are updated with the sum of both costs.

By contrast, the specialized weights for lemmas and factors (from $\phi(\sum)$ to softmax in Figure 4.2) are updated only with their specific cost.

If we apply the sum of both costs and we update the weights depending only on lemmas, we obtain Equation 4.6.

$$W^L_{t+1} = W^L_t - \lambda \frac{\partial (C^L + C^F)}{\partial W^L_t} \tag{4.6}$$

where W^L are the weights to be updated and λ is the learning rate.

When we derive the sum of the costs, we obtain Equation 4.7.

$$W^L_{t+1} = W^L_t - \lambda \left(\frac{\partial C^L}{\partial W^L} + \frac{\partial C^F}{\partial W^L} \right) \tag{4.7}$$

Consequently, the calculation of the derivatives of C^F and W^L results into 0, it does not change the learning of the network (see Equation 4.8).

$$W^L_{t+1} = W^L_t - \lambda \left(\frac{\partial C^L}{\partial W^L} + 0 \right) \tag{4.8}$$

Finally, the backprogation for the specialized weights depending on lemmas corresponds to the Equation 4.9.

$$W^L_{t+1} = W^L_t - \lambda \frac{\partial C^L}{\partial W^L} \tag{4.9}$$

Similarly, Equation 4.10 shows the weight update formula for the weights depending on factors.

$$W^F_{t+1} = W^F_t - \lambda \frac{\partial C^F}{\partial W^F} \tag{4.10}$$

This means that each output cost will be applied to its corresponding weights.

The decoder of the FNMT architecture presented in Figure 4.2 may lead to sequences with different length since lemmas and factors are simultaneously generated but in separated outputs. Indeed, each sequence of symbols ends when the end-of-sequence (`<eos>`) symbol is generated and nothing prevents the lemma generator to output the `<eos>` symbol before or after the factors generator. To avoid this scenario, the length of the factors sequence is constrained to be equal to the length of the lemma sequence. This implies 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 they are the symbols carrying most of the meaning.

4.2.3 Handling beam search with factors

The beam search (Jelinek, 1969) procedure has also been extended with respect to the original approach, since we are actually facing two beams (one for lemmas and one for factors). We need to deal with the multiple outputs because we do not want to rely solely on the lemma sequence to decide which are the best sequences. Then, we merge the two beams as shown in Figure 4.3.

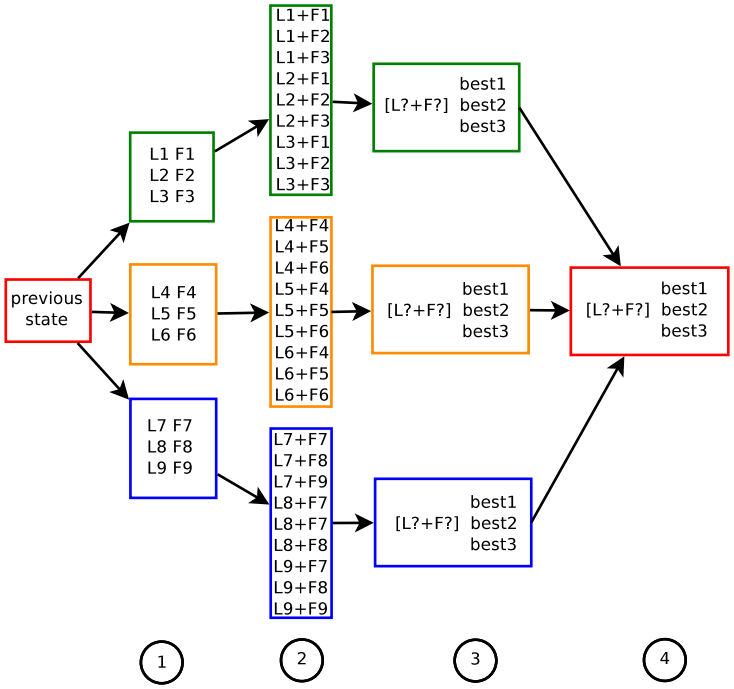

Figure 4.3: Illustration of the Factored NMT system beam search with a beam size set to 3.

Once the best lemma and factors hypotheses are generated for each partial hypothesis (stage 1), the cross product of those output spaces is performed. By this mean, each lemma hypothesis is associated with each factors hypothesis (stage 2). Note that the ideal case would be to associate lemmas with only their possible factors (this case is addressed in Section 5.1.2 checking a lookup table built with the training data which maps lemmas with their possible factors). Afterwards, we keep the k-best combinations for each sample, with k being the beam size (stage 3). Finally, the number of best hypotheses is reduced again to the beam size for further processing (stage 4).

This procedure allows our system to have one stream with two synchronized outputs. This is in contrast with multitask learning, where the different tasks are generally asynchronously processed. In our case, a strong relation links both outputs. This relation has to be taken into account in order to obtain the best performance.

4.2.4 Postprocessing: from factors to words

Once we obtain the factored representation outputs from the neural network, the post-process to fall back to the surface form is performed. This step is not trivial.

For that purpose, we built a lookup table to match the lemmas and factors as keys with the surface forms as values. This knowledge is also extracted from the MACAON tool for French language, which given a lemma and some factors, provides the word candidate.

For the sake of simplicity, the first candidate is taken for the very few cases when there are several proposals of surface forms for the same pair of lemma and factors. This is not the optimal case, it would be better to output the most frequent surface form or using a language model which takes into account the context of the word (these cases are addressed in Section 5.1). In French, in most of the cases when several words are proposed for the same pair of lemma and factors, all the proposals are correct and their choice only depends on the situation. For example, for written or spoken versions or formal or informal situations. In other cases (*e.g.* name entities) where the surface form corresponding to the lemma and factors is not found, the system outputs the lemma itself.

4.3 Preliminary experiments

After describing the FNMT baseline approach, we present a first set of experiments to evaluate the effectiveness of the FNMT system. We first explain the setup of the experiments and afterwards the different results.

4.3.1 Data processing and selection

We evaluate our approach on the English to French Spoken Language Translation task from IWSLT 2015 evaluation campaign[1]. We use the parallel corpora provided by the organisers. A preprocessing to change the written text in the source language to simulate an automatic spoken recognition tool output has been done (*e.g.* lowercasing, changing digits into letters, removing punctuation marks, etc.). The target language text has been lowercased. A data selection method (Rousseau, 2013) has been applied using the available parallel corpora (news-commentary, united-nations, europarl, wikipedia, and two crawled corpora) and Technology Entertainment Design (TED[2]) corpus as in-domain corpus. The tokenization process has been done according to Moses tokenizer. We also do a preprocessing to convert html entities and filter out the sentences with more than 50 words for both source and target languages as done in previous works (Bahdanau et al., 2014). We finally end up with a selected corpus of 2M sentences which is a small training dataset, 147k unique words for English side and 266k unique words for French side. A development dataset (dev15) composed by three datasets of previous IWSLT evaluation

[1] https://sites.google.com/site/iwsltevaluation2015
[2] https://www.ted.com

campaign years (dev10, test10 and test13) has been used for early stopping purpose in order to chose the best model. In addition, a testing data (test15) composed by two testing sets (test11 and test12) has been used to evaluate and compare the models (see Table 4.2 for more details on the data).

Data	Corpus name	Datasets	#Sents	#Unique words	
				EN	FR
Training	train15	data selection	2M	135k	212k
Development	dev15	dev10+test10+test13	3.6k	6.7k	8.5k
Testing	test15	test11+test12	1.9k	4.1k	5.2k

Table 4.2: IWSLT'15 English→French datasets statistics for preliminary experiments (lowercased).

4.3.2 Impact of factorization process

We have calculated an estimation of the loss of the process transforming words into factors during preprocess and factors into words in the postprocess. For this purpose, we have factorized the reference of the training, development and test sets for this experiment. This process consists of transforming the words into lemmas and factors. Then, we combine the extracted lemmas and factors to obtain again the words with the factorization process. Once we get the resulting words file, we compare it with the original reference file.

We computed the Word Error Rate (WER), defined in Equation 4.11 between the two files.

$$WER = \frac{Substitutions + Deletions + Insertions}{\#reference\ words} \tag{4.11}$$

Table 4.3 shows the results of WER including correct words, substitutions, deletions and insertions. The values correspond to the percentages and number of them in parenthesis.

Data	WER	Correct	Subs.	Del.	Ins.
train15	0.3% (131k)	99.7% (43M)	0.3% (131k)	0% (174)	0% (120)
dev15	0.5% (357)	99.5% (77k)	0.5% (357)	0% (0)	0% (0)
test15	1.4% (537)	98.6% (39k)	1.4% (537)	0% (0)	0% (0)

Table 4.3: WER between reference and the factorised reference reconverted in words.

When observing train15 results, the value of WER is only 0.3%. Similar results are observed for dev15 evaluation, where the value of WER is 0.5%. Finally, there is small increase of error for test15, 1.4% of WER.

We observed the differences between the two files. For example, we saw that sometimes the original testing file include the word "ça" and the post processed file has changed this word to "cela" and the reversed also occurred. Both translations are correct and they mean exactly the same but "cela" is more formal and less used. Another example is that the original file contains the word "labo" (lab in English) which is the informal and shorter form to say "laboratoire" (laboratory in English). In all these examples, both words share the same lemma and factors. This could be normalized choosing only a formal or informal way in all cases.

We can conclude that the process going from words to factors produces a small loss that can be reduced. On the other hand, the benefits of training a model in a factored level compensate this small loss.

4.3.3 Training details

For the training of all models, we used `NMTpy` (Caglayan et al., 2017), a Python toolkit based on Theano (Al-Rfou et al., 2016) and available as open-source software[3]. The `NMTpy` toolkit is an implementation of attentive Neural Machine Translation (NMT) (Bahdanau et al., 2014). For FNMT systems, I implemented new models in `NMTpy` to be able to train with multiple factors in input and output sides of the network.

The following hyperparameters have been chosen to train the systems. The embedding and recurrent layers have the dimensions 620 and 1000, respectively. The batch size is set to 80 sentences and the parameters are trained using the Adadelta optimizer (Zeiler, 2012). In order to avoid exploding gradients, we clipped the norm of the gradient to be no more than 1 (Pascanu et al., 2012). The initialization of the weights has been done with the *Xavier* (Glorot and Bengio, 2010b) method.

The validations start at the second epoch and they are performed every 5000 updates. Early stopping is based on BLEU (Papineni et al., 2002b) with a patience set to 10 (early stopping occurs after 10 evaluations without improvement in BLEU). Once the models are trained (approximately during 5 days), we set the beam size to 12, as this is the standard value for NMT (Bahdanau et al., 2014), when translating the development or testing datasets. The development dataset is used for early stopping and the testing dataset to evaluate the generalization power of the models.

The vocabulary size of the source language is set to 30k for all models. The target vocabulary size is set also to 30k for NMT models. For the sake of comparability and consistency, the same value (30k) is used for the lemma output of the FNMT system. This 30k FNMT lemma vocabulary includes 17k lemmas from the original word level NMT vocabulary because all the derived forms of the verbs, nouns, adjectives, etc. are discarded. The lemmas vocabulary is done as the word level including the most frequent tokens in the training dataset. Then, lemmas from 17k until 30k are included in the target lemma vocabulary but not in the target word vocabulary. The factors have 142 different units in their vocabulary.

[3] https://github.com/lium-lst/nmtpy

4.3.4 Generalization results

The FNMT system aims at integrating linguistic knowledge into the decoder, in order to obtain better performance when facing the restriction of the number of outputs in target side. To assess the feasibility and estimate the potential gain of our approach, we performed a set of experiments when having smaller or bigger vocabulary output size. We compare the factored level NMT approach with the standard word level NMT approach.

We varied the size of the output layer from 5k to 30k in the NMT system and the lemmas output layer in the FNMT system. The factors output vocabulary size remains in 142 in all the cases. The results are presented in Table 4.4.

	Output							Oracle	
Model	size	word voc.	(%) Cov.	# UNK	# Par.	METEOR	BLEU	METEOR	BLEU
NMT	30k	30k	97.89	924	89.7M	61.01	40.53	-	-
FNMT	30k+142	172k	99.14	435	89.8M	**62.58** (+1.57)	**41.26** (+0.73)	63.22	42.67
NMT	20k	20k	96.94	1093	77.3M	61.68	40.80	-	-
FNMT	20k+142	139k	98.77	555	77.4M	62.26 (+0.58)	41.09 (+0.29)	62.84	42.40
NMT	10k	10k	94.48	2312	64.9M	59.19	38.50	-	-
FNMT	10k+142	85k	97.66	885	64.9M	61.65 (+2.46)	40.39 (+1.89)	62.10	41.56
NMT	5k	5k	91.30	3180	58.7M	57.76	36.42	-	-
FNMT	5k+142	48k	95.70	1616	58.7M	59.87 (+2.11)	39.05 (+2.63)	60.31	40.03

Table 4.4: Comparison of the NMT and FNMT systems performance in terms of BLEU and METEOR score evaluating at word level (7 and 8 columns) with IWSLT'15 dataset. The size of the output layer and the size of the corresponding word vocabulary are presented in columns 2 and 3. Columns 4, 5 and 6 show coverage of the testing file, number of generated unknown tokens and number of parameters, respectively. Last two columns corresponds to the oracle for factors output.

The FNMT system obtains better performance compared to the NMT system in terms of word level BLEU and METEOR (Lavie and Agarwal, 2007) score in all the cases. If we first focus in the first 2 rows using 30k vocabulary size, the difference in METEOR between the two systems is 1.57 and in terms of BLEU score is 0.73. METEOR metric supports stem match and paraphrasing for French language and its evaluation can benefit the FNMT systems. When we observe 20k output layer size, the difference between the performance of the two systems is reduced (0.58 METEOR points and 0.29 BLEU points). This lower value might means that NMT system is better trained with 20k target vocabulary size and the data is sparse to use 30k target vocabulary size. When reducing the output layer size (10k and 5k), we see the power of FNMT system, the performance difference between the two systems is even higher (2.46 METEOR points and 1.89 BLEU points using 10k setup and 2.63 METEOR points and 2.11 BLEU points using 5k setup). NMT system does not have enough coverage to obtain good results using 10k or 5k target vocabulary sizes.

In order to estimate the boundaries of FNMT model, we computed the oracle for factors output which corresponds to ignore the errors caused by the factors, *i.e.* if

we produce the correct lemma (we use the factors reference to correct the factors output), then the correct word is generated (see last two columns of Table 4.4). We can see that a potential gain of more than 1.4 BLEU points and 0.64 METEOR points using 30k output lemma vocabulary size can be achieved with a perfect modeling of the factors, which is encouraging. If we were able to improve factors output, the differences between the two models would be even larger.

As Table 4.4 illustrates, the FNMT approach is able to model a bigger word vocabulary while preserving manageable output layers sizes. This is due to the fact that the factorization process is able to generate words which are unseen in the training corpus, augmenting the expressiveness of our model. For the sake of comparison, we provide the target vocabulary size for the standard NMT and the FNMT systems. For example, with an output layer size of 30k, the NMT system can model 30k words against 172k words for the FNMT system. This is an almost 6 times larger word vocabulary. In the smallest output layer size (5k setup), the FNMT system can still handle 48k words which is still big enough size.

One consequence is that the word coverage is higher for the FNMT than for the NMT system, as shown in column 4. However, for the first two systems (first two rows), we see that the difference of the coverage is small. When decreasing the output layer size, we can observe that the coverage decreases slowly for FNMT systems compared to the word-based system. The FNMT approach performs better and better than standard NMT when the coverage difference becomes higher. This proves that the approach is sound and well performing, when dealing with smaller vocabulary size. This is of course dependent on the linguistic knowledge available in the factorization process. This is exactly the sought behavior: by integrating *a priori* linguistic knowledge, we reduce the impact of the training conditions (domain, data availability, etc.) on the performance of the system.

We measured also the number of unknown words (#`UNK`) generated by the systems. The #`UNK` produced by FNMT system are the half of the ones produced by NMT system, which is a big reduction. In those experiments, we did not use any specific method to replace them (*e.g.* put source words aligned to them, use a dictionary, etc.)

Moreover, the number of parameters to train also decreases according to the size of the output layer, as shown in column 6, allowing a simpler training because we have to learn less weights in the model. For example, using a layer size of 10k instead of 30k (3 times smaller) for factored model, we obtain a small drop of 0.87 points in BLEU. By contrast, in the NMT base model we observe a drop of 2 points in BLEU comparing the same output sizes 30k and 10k. We see that the FNMT model can generalize better than the NMT model in low resource conditions. FNMT system does not require that much parameters as NMT system to obtain the same performance. For example, FNMT system using 20k or 10k output sizes obtains better performance than NMT system using 30k output size.

In this experiment, we have seen the utility of FNMT system getting improvements in translation performance. When reducing the output layer size, the gains are much bigger using FNMT system than NMT system.

Translation examples of FNMT system performance

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.

1	Src	set of adaptive choices that our lineage made
	Ref	de choix adaptés établis par notre lignée
	NMT/+UR	de choix UNK/adaptif que notre UNK/lignage 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/+UR	enzymes qui les UNK/réparions et les UNK/celui-.
	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/+UR	santa UNK/marta dans le nord de la colombie
	FNMT	santa marta dans le nord de la colombie

Table 4.5: Examples of sentences translated by NMT and FNMT systems using IWSLT'15 dataset. UR corresponds to the output of the unknown replacement method.

The translation examples in Table 4.5 show the FNMT system outputs and the NMT system outputs when using 30k vocabulary size. First example shows that the FNMT 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 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 training dataset at word level being in the position 34642 of the most frequent words. As a consequence, it is not sufficiently frequent to include it in the shortlist of the NMT system (the vocabulary size is only 30k). On the other hand, the lemma 'réparer' appears 8523 times in the training dataset at lemma level (tokens represented with their lemma) being in the position 2480 of the most frequent lemmas. Therefore, it is included in the lemmas shortlist and we are able to generate 'réparent' from the lemma and factors outputs (verb in present tense and third person in plural). Moreover, the verb in English 'put together' is translated to 'assemblent' in the reference which is a synonym of the

FNMT system translation 'mettent ensemble'.

Example 3 shows an FNMT system translation performing as the reference. We are able to generate the name entity 'marta' that is not in the NMT system vocabulary. This is due to the fact that FNMT models directly the lemmas in the softmax and can cover a larger set of lemmas than the NMT model which has the surface forms in the softmax. These examples show the potential of the FNMT system generating new words and reducing the number of unknown words.

4.3.5 Comparative results

For the sake of comparison with the state of the art systems, we have trained BPE models. We applied the subwords units at the output side as the factors are only applied in target language with FNMT approach. We set the number of merge operations for BPE algorithm, as explained in Sennrich et al. (2016c), following Equation 4.12.

$$\#merge\ ops = vocab.\ size\ - \#characters \tag{4.12}$$

Moreover, we have trained multiway, multilingual NMT models (Firat et al., 2017). 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 postprocessing as done with the FNMT system. BPE and multilingual models have been compared to NMT and FNMT systems.

The vocabulary size used for this experiment is 30k for source language (English) and target language (French words and French lemmas). The vocabulary size for French factors is still 142.

	METEOR	**BLEU**			
Model	**word**	**word**	**lemma**	**factors**	**#UNK**
NMT	61.01	40.53	42.77	51.39	924
BPE	61.31	41.07	43.43	51.79	0
Multilingual	57.90	34.35	43.31	48.25	472
FNMT	**62.58**	**41.26**	42.53	45.26	435

Table 4.6: Comparison of the NMT, BPE, multilingual and FNMT systems performance in terms of METEOR score at word level (column 2) and BLEU score at word level, and separately, each output lemma and factors (columns 3, 4 and 5, respectively). Column 6 corresponds to the number of generated unknown words. The dataset used is IWSLT'15 and the vocabulary size is 30k.

The FNMT system obtains better performance compared to the state of the art systems in terms of word level BLEU and METEOR score (see Table 4.6).

We also evaluated each output (lemmas and factors) separately in terms of BLEU for the FNMT and the multiway, multilingual system. We decomposed the reference in lemmas and factors as we do in preprocessing of the dataset. Then, we compare lemmas and factors outputs with lemma and factors reference, respectively. The translation output of NMT and BPE systems have been decomposed in lemma and factors as we do in preprocesssing. Then, we compare the extracted lemmas and factors from the translation outputs with the references at both levels as well.

We already have commented the results at word level for NMT system in previous experiment. When evaluating at lemma level, we see a slight improvement compared to the FNMT system. At factors level, NMT system obtained a big improvement compared to FNMT system. It is worth to mention that the lemmas and factors from NMT and BPE systems have been artificially extracted directly from the words output (words→factors), as opposite to FNMT and multilingual lemmas and factors outputs which are generated by the network.

BPE obtains the best performance at lemma and factors levels in terms of BLEU. However, we can see that BPE model performs worse in BLEU (-0.19) and METEOR (-1.27) scores at word level than FNMT system. METEOR metric can capture better the morphology than BLEU metric, which makes the performance difference of BPE and FNMT system bigger. The worse performance of BPE system compared to FNMT system can be because incorrect words can be produced when merging BPE units. Other reason could be because BPE method is only applied on one side and the best performance of BPE method is obtained when is applied on both sides.

Multilingual approach performance at word level is the lowest. However, we can observe that multilingual system gets higher results than FNMT when evaluating the two outputs independently. This might be due to the desyncronization of the two outputs which are trained independently. On the other hand, when constructing the words from the factors, the multilingual system obtains a much lower score (almost 7 BLEU and 4.5 METEOR points less) than the FNMT system. This is due to the difficulty of combining both lemmas and factors when they are generated independently. By contrast, our FNMT system can generate the two outputs with a strong correlation making it possible to build the words correctly.

Moreover, our FNMT system produces less number of unknown words (#`UNK`) than NMT and Multilingual systems. This tends to prove that the FNMT system effectively succeeds in modeling more words compared to the word-based NMT system. BPE system does not generate `UNK` because all are encoded as BPE units. However, this is not reflected in BLEU and METEOR scores at word level, the lack of `UNK` in BPE system does not mean that the translation is always correct.

4.3.6 Evaluating each output

In order to better evaluate the different outputs of our FNMT system, we trained three different standard NMT models. The three NMT models have the same source (English words) but each of them has a different output: (1) French words, (2) French lemmas and (3) French factors. We compare the values of those three NMT

models with the outputs of the Factored NMT system which models lemmas and factors at the same time. This allows us to estimate the impact of jointly modeling lemmas and factors in the same architecture. The results are presented in Table 4.7.

	METEOR			BLEU		
Model	**word**	**lemma**	**factors**	**word**	**lemma**	**factors**
NMT	61.01	61.97	64.93	40.53	42.07	47.67
FNMT	**62.58**	61.89	62.44	**41.26**	42.53	45.26

Table 4.7: Comparison of the performances between standard NMT system and the Factored NMT system in terms of METEOR and BLEU computed at word, lemma and factors level. The first line corresponds to 3 standard NMT systems built to generate, respectively, words, lemmas and factors at its output. The dataset used is IWSLT'15 and the vocabulary size is 30k.

As already seen in previous presented experiment, the FNMT system obtains better word level scores compared to standard NMT.

When producing lemmas, the BLEU score is higher (+0.46) for FNMT system than NMT system and similar in METEOR score. This can be due to the fact that FNMT model contains also the factors information which may help the lemmas generation.

However, the opposite case occurs when evaluating factors separately. The NMT system yields +2.41 BLEU points and +2.49 METEOR points compared to FNMT system which is a big difference. This tends to suggest that there is room for improvement in our architecture, and we could obtain better results by better modeling the factors.

However, the BLEU scores for factors in both systems are pretty low when considering that the output layer size is only 142. This can be due to two different causes. First, the neural network is not able to correctly model this small output. The lemma output has an increased number of specialized weights, while factors output has a small number of parameters. Additionally, there is a big difference between the output sizes, and the gradients coming from lemmas and factors might be of different scales. Second cause can be that English words to French factors translation task is complex. A monolingual PoS tagging task from English words to English factors is easy but our objective is the French PoS tagging. English is a grammatically poor language. Therefore, some factors like gender in nouns or person in verbs are missing and those factors are necessary in French language.

Also, we have to take into account the fact that we are giving more priority to the length of the lemmas sequence than the factors one sequence during beam search. In addition, the feedback of the model on those experiments is only the information coming from lemmas. The next set of experiments has been carried out to improve the factors generation in order to find a better setup of FNMT system.

4.4 Design of the FNMT architecture

We designed different options in the architecture in order to obtain the best version of the FNMT system. We observed previously that the factors output performance can be improved. Therefore, we modified some settings of the architecture in order to get a better factors output. First of all, some experiments have been performed modifying the architecture by adding links between the two outputs. Then, we tested different options for the feedback of the model. The vocabulary size used for these experiments is 30k as previously.

4.4.1 Dependency models to improve factors prediction

In the previous experiment, we saw that the lemma score performs similarly for FNMT and NMT systems. Nevertheless, the difference between the two systems is big when evaluating the factors. We can think that the prediction of factors should be an easy 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.

Chain of NMT systems

The first experiment we performed to model the dependency consists of a chain of two standard NMT systems.

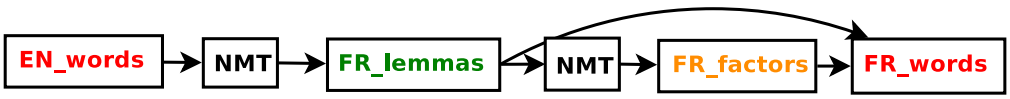

Figure 4.4: Chain of NMT systems schema

Figure 4.4 shows a schema of how the chain of NMT systems is trained. The first NMT system has as input the English source words as usual, and as output the French target lemmas. Then, the second NMT system translates from the French target lemmas generated in the first system to the French target factors. Finally, we obtain the final French target words combining the French target lemmas generated in the first system and the French target factors generated in the second system.

Table 4.8 shows in the first line the results in BLEU score of the FNMT system at word, lemma and factors levels. The second line shows the chain of NMT systems results at each step. The first NMT system ($\text{EN}_{words} - \text{FR}_{lemmas}$) of the chain which is evaluated at lemma level performs almost 0.5 BLEU points worse than FNMT system. As expected, the score of the output of the second system $\text{FR}_{lemmas} - \text{FR}_{factors}$ evaluated at factors level is much higher (80.67 BLEU points) than FNMT system due to the easy task of translating from lemma to factors on the same language. Nevertheless, the *factors-to-word* process result extracted from the outputs of the two NMT systems is much worse score than FNMT system. This can be due to the difficulty of *factors-to-word* process to handle the asynchronous

	BLEU		
Model	**word**	**lemma**	**factors**
FNMT	**41.26**	42.53	45.26
	factors-to-word	$\text{EN}_{words} - \text{FR}_{lemmas}$	$\text{FR}_{lemmas} - \text{FR}_{factors}$
Chain NMT	39.72	42.07	**80.67**

Table 4.8: Comparison of the performances between the chain of NMT systems and the FNMT system in terms of BLEU computed at word, lemma and factors level. The first line corresponds to the results on the FNMT system at the three levels. The last two columns in the second line corresponds to the evaluation of the outputs of two standard NMT systems trained with the sequence pair $\text{EN}_{words} - \text{FR}_{lemmas}$ and $\text{FR}_{lemmas} - \text{FR}_{factors}$, respectively. The column about words in the second line is obtained from the combination of the last two columns in the same line (lemmas and factors). The dataset used is IWSLT'15.

outputs of both systems trained separately, and having different alignments between the source and target words. This high performance gives us an idea of creating a dependency from lemmas to factors, in order to help the factors output to produce better sequences.

Lemma dependency

One observation that can be made is that while generating factors seems easier due to the small number of the possible outputs (only 142), the BLEU score is not as high as what we could expect in the FNMT system. 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 factors 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 on 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 4.5) which projects the lemma embeddings into the hidden layer used to generate the factors. We use as feedback of the model only the previous lemma as in previous experiments.

We have implemented two possibilities for the lemma dependency model:

1. The embedding of the previous lemma to provide recent context.

2. The embedding of the current lemma corresponding to the factors to be generated. For this option, we have also modified the beam search function. The lemmas are first generated and the factors are generated in a second step based on the generated lemmas.

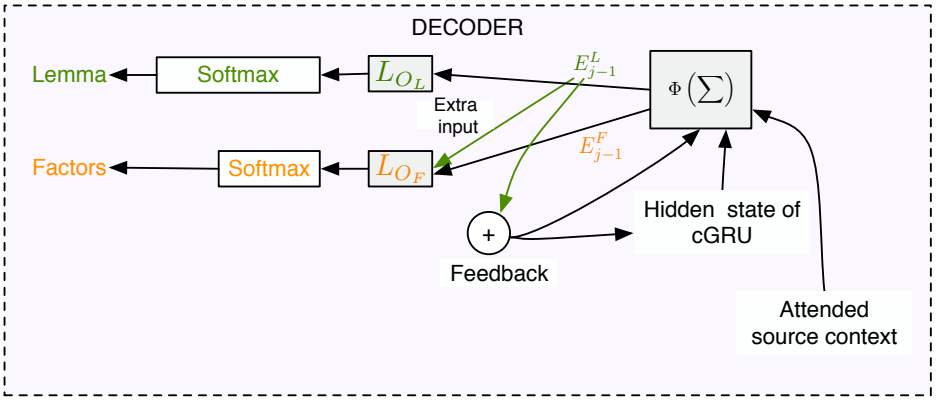

Figure 4.5: Lemma dependency model decoder detail

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 the embedding of them as feedback to the factors generation. This is like including a language model for factors (see Figure 4.6).

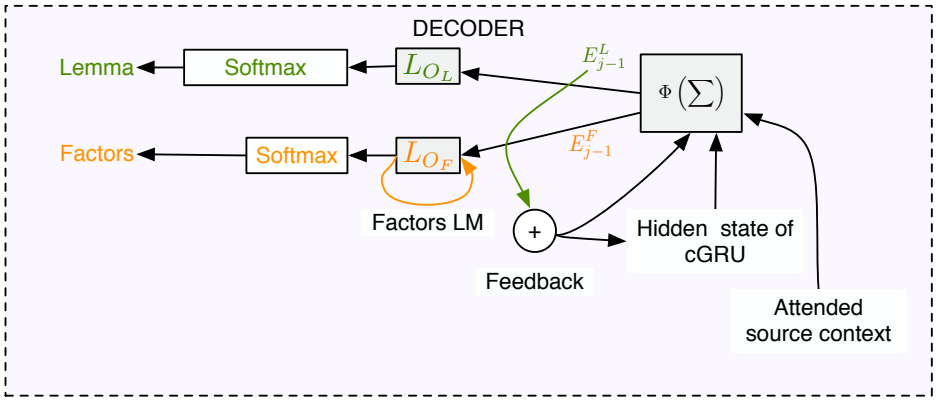

Figure 4.6: Factors dependency model decoder detail

The results applying the dependency models techniques are presented in Table 4.9.

We observe that the dependency model using the previous lemma (FNMT2) does not improve the results in comparison to the FNMT model without dependency (FNMT1). This can be due to the fact that we have already given the previous lemma as main feedback for the recurrent hidden state.

Moreover, the dependency model using the current lemma (FNMT3) does not improve either the performance of the FNMT1 in terms of BLEU and METEOR scores on words. If we look at the BLEU scores at lemma and factors levels, the

Model	Dependency	METEOR word	BLEU word	BLEU lemma	BLEU factors	#UNK
FNMT1	-	62.58	41.26	42.53	45.26	435
FNMT2	previous lemma	60.64	40.52	41.58	44.52	424
FNMT3	current lemma	60.59	39.96	40.85	44.92	361
FNMT4	previous factors	62.38	40.68	41.95	45.56	439

Table 4.9: Performance in terms of METEOR computed at word level, BLEU computed at word, lemma and factors level and the number of generated unknown words of the FNMT system when using different dependency options. The dataset used is IWSLT'15.

scores are also lower using FNMT3 than FNMT1 model. However, the factors generation using FNMT3 improved compared to FNMT2. As opposite, FNMT2 generates better lemmas than FNMT3. Therefore, previous lemma dependency helps to generate lemmas and current lemma dependency helps to generate factors. FNMT3 scores are worse than FNMT1, the current lemma dependency did not give an improvement of the performance. Possibly, in other morphologically richer languages than French which include even more factors, the lemma dependency would help to predict a better translation. In addition, FNMT3 generates the lowest number of `UNK` compared to the rest of the models. `UNK` is only produced when a lemma is unknown. The lowest #`UNK` can be explained by the fact that the lemma is generated first, before factors and factors generation contains a direct input of the current lemma embedding. This fact can give more priority to the lemma generation, avoiding `UNK`.

The last model using previous factors dependency (FNMT4) to feed the factors output, decreases the BLEU and METEOR values at word level and BLEU score at lemma level compared to FNMT1. On the other hand, the score at the factors level is higher than FNMT1, here we see the usefulness of factors dependency. The previous factors dependency allows the model to use generated factors for feedback and to act as a factors language model to improve its output. Unfortunately, this factors output improvement is not reflected at the word level. FNMT4 differs from FNMT1 only by having a small extra information due to the small size of factors output layer. Therefore, the full network is not affected by this.

Current lemma dependency model translation example

Besides the worse results for FNMT model using the current lemma dependency than simple FNMT model without dependency, we have found some examples where we can see the benefits of having this dependency in FNMT system to help the generation of better factors output.

Table 4.10 shows one example where the FNMT without dependency generates a wrong factors output. Simple FNMT produces the lemma "être" and its correspond-

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	personne	ne	sait	ce	que	nous	faisons		.
Lemma	dependency	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 4.10: Examples of translations with FNMT and FNMT with current lemma dependency. The dataset used is IWSLT'15.

ing factors are "cln-3-s" (clitic nominative, third person of the singular), which are not correct. On the contrary, FNMT including current lemma dependency produces correctly the lemmas "nous faire" with their factors "cln-1-p" (clitic nominative, first person of the plural) and "v-P-1-p" (verb in present tense, first person of the plural), respectively. This example illustrates the effectiveness of the dependency method.

Translation of different sentence lengths

Translating long sentences is a challenge in machine translation because the context is long and the possibility to produce errors translating all the words in the sentence is high. Then, we would like to see the benefits of FNMT system facing this challenge. In this experiment, we compare the standard NMT system and the FNMT system providing previous factors dependency when translating different sentence lengths. For this purpose, we evaluated the translation performance clustering the sentences with respect to their source sentence length. Figure 4.7 shows the BLEU score for sentences between 10 and 100 words with intervals of 10.

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 when translating long sentences (between +1.5 and 3 BLEU points). This improvement may be due to the less sparsity on the lemma and factors space. The FNMT system is not as sensitive as NMT system to the sentence length because FNMT system makes easier choices translating directly to lemmas instead of words. The drop observed at interval 70 is due to the fact that at this many sentences with this length perform bad.

4.4.2 Feedback of the model

FNMT architecture generates two outputs at the same time, therefore, we need to redefine what kind of feedback is more suitable according to the double output information. Several solutions are possible to use either or both embeddings as feedback.

We have already described the conditional GRU used in the decoder of our system (see Section 2.2). The conditional GRU is composed of two GRUs, the first GRU (GRU_1) cell of the decoder is fed by its previous hidden state and the feedback (*i.e.* the previous generated symbol). The following formulation redefines the GRU_1

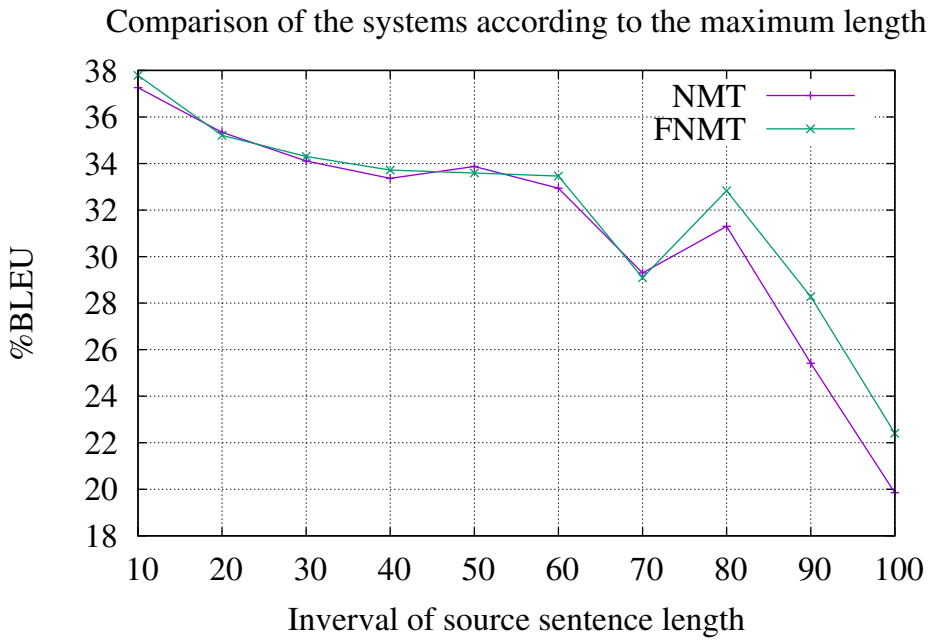

Figure 4.7: Comparison in terms of BLEU of the NMT and FNMT with factors dependency systems according to the maximum source sentence length. The dataset used is IWSLT'15.

for the FNMT system.

$$\mathbf{s}'_j = \text{GRU}_1\left(\mathbf{fb}(y_{j-1}), \mathbf{s}_{j-1}\right) \tag{4.13}$$

where **fb** is the function which computes the feedback from the previous output y_{j-1} instead of $\mathbf{E}$ in the standard NMT system. The rest of the symbols remain the same as previous.

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 4.14).

$$\mathbf{fb}(y_{t-1}) \quad = \quad \mathbf{E}[y^L_{t-1}] \tag{4.14}$$

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

For the sake of comparison, in the same direction but with the other output information, we used only the factors embedding as feedback (see Equation 4.15).

$$\mathbf{fb}(y_{t-1}) \quad = \quad \mathbf{E}[y^F_{t-1}] \tag{4.15}$$

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

Another straightforward operation is summing the embeddings (technique used in Mikolov et al. (2013)) of the previous lemma and previous factors, as described in Equation 4.16.

$$\mathbf{fb}(y_{t-1}) \quad = \quad \mathbf{E}[y^L_{t-1}] + \mathbf{E}[y^F_{t-1}] \tag{4.16}$$

While this could seem unnatural, by doing this, we hope to obtain a joint vector representation of both the lemma and the factors constraint to have the same size as using only one of them.

We investigated whether the neural network can learn a better combination of the lemmas and factors embeddings using a linear (Equation 4.17) or non-linear (Equation 4.18) operation instead of a simple sum.

$$\mathbf{fb}(y_{t-1}) = (\mathbf{E}[y^L_{t-1}] + \mathbf{E}[y^F_{t-1}]) \cdot W_{fb} \quad (4.17)$$

$$\mathbf{fb}(y_{t-1}) = \tanh\left((\mathbf{E}[y^L_{t-1}] + \mathbf{E}[y^F_{t-1}]) \cdot W_{fb}\right) \quad (4.18)$$

In this case, W_{fb} are trained weights.

In addition, we used the concatenation of both target embeddings as feedback using a linear (Equation 4.19) or non-linear (Equation 4.20) 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 but increasing model size (#parameters).

$$\mathbf{fb}(y_{t-1}) = [\mathbf{E}[y^L_{t-1}]; \mathbf{E}[y^F_{t-1}]] \cdot W_{fb} \quad (4.19)$$

$$\mathbf{fb}(y_{t-1}) = \tanh\left([\mathbf{E}[y^L_{t-1}]; \mathbf{E}[y^F_{t-1}]] \cdot W_{fb}\right) \quad (4.20)$$

		METEOR	BLEU		
Model	**Feedback**	**word**	**word**	**lemma**	**factors**
FNMT1	Lemma	62.58	41.26	42.53	45.26
FNMT2	Factors	57.88	36.95	37.94	47.02
FNMT3	Sum	61.52	40.74	41.37	46.48
FNMT4	Linear Sum	60.97	40.52	41.34	46.84
FNMT5	Tanh Sum	60.06	40.10	40.93	46.20
FNMT6	Linear Concat	61.15	40.51	41.39	46.86
FNMT7	Tanh Concat	61.88	40.60	41.45	47.07

Table 4.11: Performance in terms of METEOR on word and BLEU computed on word, lemma and factors of the FNMT system when using different output embedding combinations as feedback. The dataset used is IWSLT'15.

Table 4.11 presents the results obtained with systems integrating the different output embedding combinations as feedback.

As expected, when using only lemma as feedback (FNMT1), the system better estimates the lemmas probabilities. As a consequence, there is a significant reduction of the performance on factors when comparing with the rest of the models. Lemma embedding contains the main information of the word. Therefore, the model using the lemma as feedback gets better scores at word level than the rest of the models.

Then, we trained models to study how we can include the factors embedding for feedback to improve the performance. Firstly, we explored the possibility of having only factors embedding as feedback (FNMT2). We observe that using factors

embedding option, factors level BLEU score is one of the highest of the FNMT systems. By contrast, lemma BLEU is the lowest (more than 3 points less compared to FNMT1) and this has a great impact at word level performance (3 BLEU points less and almost 5 METEOR points less than FNMT1).

Secondly, we performed the sum of the embeddings for feedback of the model. The simple sum of the embeddings (FNMT3) gives higher scores than using its linear (FNMT4) and tanh operation (FNMT5) excepting at factors level where FNMT4 obtains higher BLEU.

In addition, we have experimented the concatenation of the two outputs embeddings to learn more parameters as feedback with a better combination of them. The linear transformation of the embeddings concatenation (FNMT6) performs similar to FNMT3 at all the levels. By contrast, the tanh transformation of the embeddings concatenation (FNMT7) gets an improvement in terms of METEOR compared to FNMT3 and FNMT6. When comparing in terms of BLEU, word level and lemma level perform quite similar to FNMT3. On the other hand, at factors level, FNMT7 gets the highest score compared to the rest of the models but it is not reflected at word level. 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. Despite summing the two embeddings contains the information of both outputs, the concatenation of the two embeddings better represents them.

4.5 Going further into FNMT system

For a better evaluation of the FNMT system, we use case sensitive data in order to have a more realistic task and increase the difficulty of the translation process having more vocabulary. We can study the advantages of the FNMT system in this environment. In order to make it possible, we have introduced an additional factor about the case information (lowercase, uppercase or in capitals). Introducing words with capital letters, standard NMT system has to keep them as different words in the vocabulary. By contrast, FNMT system can keep those words in the same lemma representation as their homologous in lowercase by adding only a new factor. Therefore, FNMT system can incorporate in the vocabulary even more words than NMT system when translating case sensitive text. We have also improved the tokenization process taking into account MACAON splitting and not just the tokenization provided by Moses. Moreover, we applied an unknown words (`UNK`) replacement technique using the alignments of the attention mechanism to replace the generated unknown words in target side. For that, we make use of an unigram dictionary automatically built with the same training dataset matching the most frequent unigrams in order to find the translation of the source word corresponding to the generated `UNK`.

The feedback embeddings (input to the decoder) and the output embeddings are tied (Press and Wolf, 2016) to enforce learning a single target representation and decrease the number of total parameters. For the sake of simplicity, we use the

concatenation of the lemma and factors embeddings as feedback of the model.

We performed a set of experiments for FNMT system and compared it with the word-based and BPE-based NMT systems.

Additionally, an analysis has been performed on some examples from NMT and FNMT systems to observe the behaviour of the alignments produced by the attention mechanism.

Finally, we applied factors at input side of the network to test if FNMT system can improve its performance applying factors on both sides.

Data processing

The IWSLT'15 dataset has been used as well for the next set of experiments but the preprocessing of the data has changed. The data selection, conversion of html entities and filtering of the data remain the same (only sentences of maximum 50 words are kept). For tokenization, we improved the procedure taking into account some differences between Moses tokenization and MACAON tokenization in order to match all the lemmas in the lookup table to the tokenized data. Then, 2M training sentences remain for training dataset. However, we kept the source language written text without any spoken text simulation. Therefore, we kept the punctuation marks, digits and case information. In the target language, we also kept the case information. As we do the experiments with case sensitive data, the number of unique words for English and French have increased. Also, the unique words for lemmas in French are 145k and the factors vocabulary increased to 357 with the addition of the new factors. The details about training dataset, development and test sets are shown in Table 4.12.

Data	**Corpus name**	**Datasets**	**#Sents**	**#Unique words**	
				EN	**FR**
Training	train15	data selection	2M	213k	233k
Development	dev15	dev10+test10+test13	3.6k	7.3k	8.9k
Testing	test15	test11+test12	1.9k	4.5k	5.4k

Table 4.12: IWSLT'15 English→French datasets statistics for further experiments (case sensitive).

Training details

Most of the hyperparameters have not changed from the experiments in previous sections 4.3 and 4.4. The models are trained with `NMTpy`. The systems are validated on dev15 dataset. The vocabulary size of the source and target language remains set to 30k.

For BPE systems, bilingual vocabulary has been built using source and target language applying the joint vocabulary BPE approach (see Section 3.1.6 for more

details). In order to create comparable BPE systems, we set the number of merge operations for the BPE algorithm (the only hyperparameter of the method) as 30k minus the number of character according to the paper (Sennrich et al., 2016c). Then, we apply a total of 29388 merge operations to learn the BPE models on the training and validation sets. During the decoding process, we use a beam size of 12 as used in (Bahdanau et al., 2014).

4.5.1 Quantitative results

The Factored NMT system aims at integrating linguistic knowledge into the decoder in order to overcome the restriction of having a large vocabulary at target side. We first compare our system with the standard word level NMT system. For the sake of comparison with state of the art systems, we have built a subword level system using the BPE method. The results are measured with two automatic metrics, the most common metric for machine translation BLEU and METEOR. We evaluate on test15 dataset from the IWSLT 2015 campaign and results are presented in Table 4.13.

	METEOR↑	**BLEU↑**			
Model	**word**	**word**	**lemma**	**factors**	**#UNK**
NMT / +UR	62.21 / 63.38	41.80 / 42.74	45.10	51.80	1111 / 0
BPE	62.87	42.37	45.96	53.31	0
FNMT / +UR	**64.10 / 64.98**	**43.42 / 44.15**	**47.18**	**54.24**	604 / 0

Table 4.13: Results on IWSLT test15. BLEU at word, lemma and factors levels and METEOR at word level performance of NMT and FNMT systems with and without `UNK` replacement (UR) are presented. BPE system results are also included. For each system we provide the number of generated `UNK` tokens in the last column.

As we can see from the Table 4.13, the FNMT system obtains better BLEU and METEOR scores compared to the state of the art NMT and BPE systems. An improvement of about **1** BLEU point is achieved compared to the best baseline system (BPE). This improvement is even bigger (**1.4** BLEU point) when `UNK` replacement is applied to both systems. In a quest to better understand the reasons of this improvement, we also computed the BLEU scores of each output level (lemmas and factors) for FNMT. The lemma and factors scores of NMT and BPE systems are obtained through a transformation of their word level output into lemma and factors. We observe yet again that FNMT systems gives better score at both lemma and factors level. Replacement of unknown words has been performed using the alignments extracted from the attention mechanism. We have replaced the generated `UNK` tokens by the translation using a dictionary of the source word which has the highest probability to be aligned to the `UNK` token (the alignment information is extracted from the attention mechanism). We see an improvement of around 1

BLEU point in both NMT and FNMT systems applying UNK replacement.

The last column of Table 4.13 shows, for each system, the number of generated UNK tokens. FNMT system produces half of the UNK tokens compared to the word-based NMT system. This tends to prove that the FNMT system effectively succeed in modelling more words compared to the word-based NMT system augmenting the generalization power of our model and preserving manageable output layer sizes. Though we can see that BPE system does not produce UNK tokens, this is not reflected in the scores. Indeed, this can be due to the possibility of incorrect words generation using BPE units in contrast to the FNMT system.

Notice that these results are better for all systems than the ones in Table 4.6 in Section 4.3.5, mainly because the data used in previous experiments did not contain punctuation marks in source language. In contrast, the data used in this experiment contain punctuation marks in both languages and they can be easily aligned helping to the translation performance. In order to justify this, we deleted the punctuation marks and lowercased the reference and the translation output of the NMT system in this section and Section 4.3.5. We computed BLEU comparing the reference and the translations with the purposed of verifying that punctuation marks help. We obtained 41.33 BLEU points for the NMT system in this section and 41.30 BLEU points for the NMT system in Section 4.3.5 which are similar results. This means that the systems are equivalent and the better results in this section is due to the better prediction of punctuation marks because they are explicitly in the source language. Therefore, the results are not comparable because of the differences in the preprocessing of the data.

4.5.2 Qualitative analysis

The strengths and weaknesses of FNMT are considered under a qualitative analysis. Translation outputs are observed and few examples are presented in Figures 4.8, 4.9, 4.10 and 4.11. NMT (top) and FNMT (bottom) outputs are presented with word level alignments.

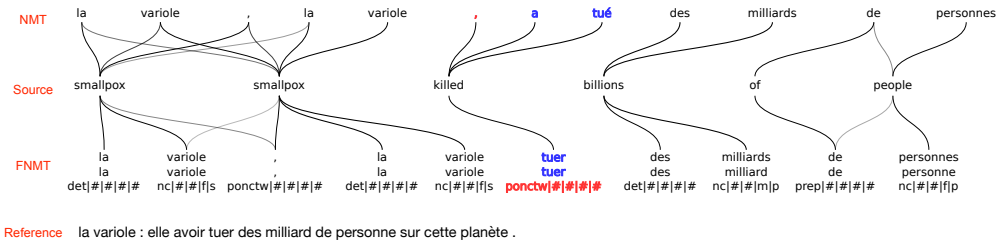

Figure 4.8: First example comparing NMT (top) and FNMT (bottom) aligned against the source sentence (middle). The FNMT system output produces wrong factors leading to an erroneous word sequence. Reference sentence corresponds to the last line. The dataset used is IWSLT'15.

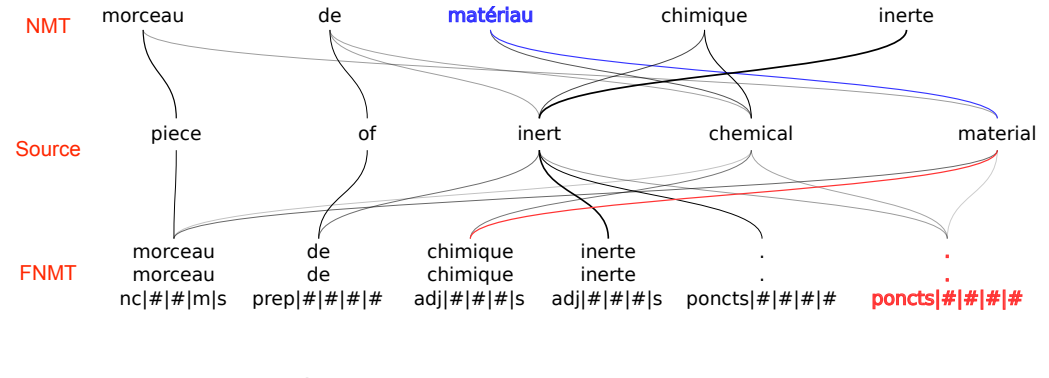

Figure 4.9: Second example comparing NMT (top) and FNMT (bottom) aligned against the source sentence (middle). The FNMT system output produces wrong factors leading to an erroneous word sequence. Reference sentence corresponds to the last line. The dataset used is IWSLT'15.

A typical problem encountered with factored NMT is shown in the Figure 4.8. The system proposed the correct lemma, but the factors are wrong and nonsense. Notice that in this case, our strategy is simple and straightforward: if no word can be generated from the lemma and the factors, then the system outputs the lemma. This issue may be overcome by improving the decoder, and more particularly the generation of the factors. One possible approach would be to constrain the factors to values which can actually be associated with the corresponding lemma. This aspect will be addressed in Section 5.1.2.

The second example in Figure 4.9 illustrates an example where the FNMT system alignment of the word *"material"* matches with the word *"chimique"* which is wrong and the translation of this word was missing. This also causes that FNMT system outputs two final punctuation symbols at the end of the sentence.

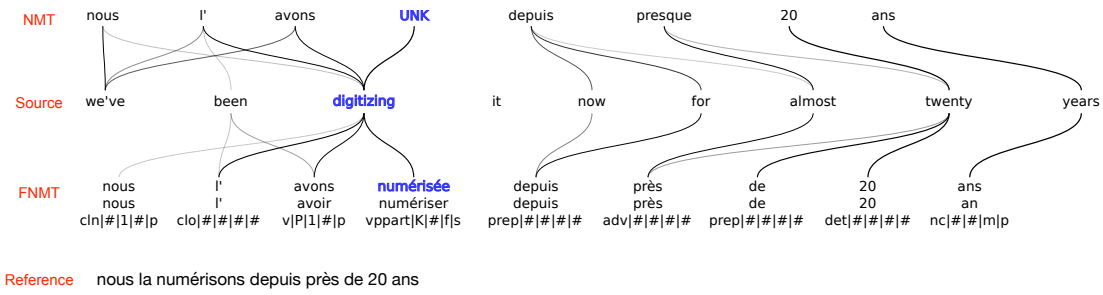

Figure 4.10: Third example of NMT (top) and FNMT (bottom) outputs aligned against the source sentence (middle). Reference sentence corresponds to the last line. The dataset used is IWSLT'15.

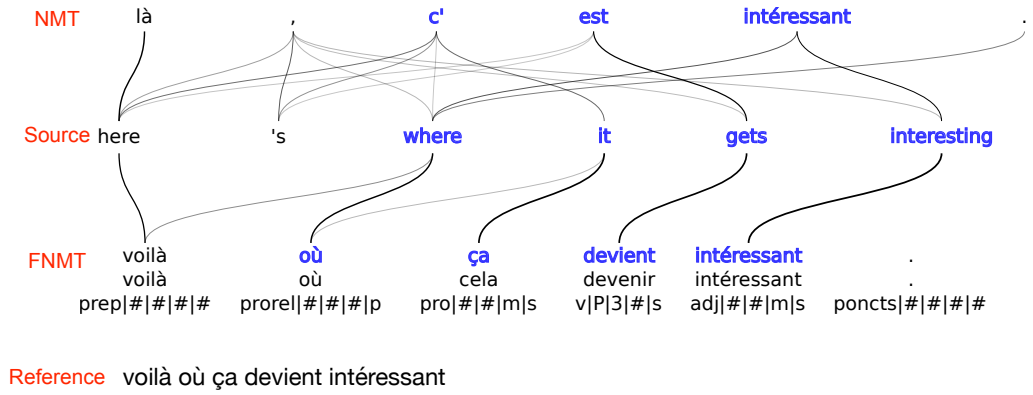

Figure 4.11: Forth example of NMT (top) and FNMT (bottom) outputs aligned against the source sentence (middle). Reference sentence corresponds to the last line. The dataset used is IWSLT'15.

Figures 4.10 and 4.11 show the outputs for two sentences which outlines the strength of the FNMT system. Figure 4.10 example has the following reference sentence: *"nous la numérisons depuis maintenant près de 20 ans . "* In the first sentence, the NMT system translates the word *"digitizing"* by the unknown (`UNK`) token. The restriction of the size of the vocabulary did not include this word in the NMT system.

On the other hand, the FNMT system can correctly translate the word *digitizing* to *"numérisée"*. This is possible since the lemma *"numériser"* is known by the system and the factors (verb in past participle, feminine and singular) are correctly generated. However, the reference translation shows the present form of the same verb (*"numérisons"*) and this is considered erroneous from BLEU point of view since we are evaluating using a single reference. Moreover, this could explain the higher improvements when output is evaluated using METEOR score. This example shows the effectiveness of our FNMT system to increase the coverage and decrease the number of unknown words.

The reference translation of the source sentence presented in Figure 4.11 is *"voilà où ça devient intéressant"*. As we can see, contrary to the baseline NMT system, the FNMT system matches exactly the reference and thus produces the correct translation. An additional interesting observation is that the alignment provided by the attention mechanism seems to be better defined and more helpful when using factors. Also, one can notice the difference between the attention distributions made by the systems over the source sentence. The NMT system first translated *"here"* into *"là"*, added a coma, and then was in trouble for translating the rest of the sentence, which is reflected by the rather fuzzy attention weights. The FNMT system had better attention distribution over of the source sentence in this case.

Comparing systems with BPE method

We have compared as well the translation outputs of NMT at word-level and at BPE-level with the ones of FNMT systems. Table 4.14 presents an example of the three systems.

Src	we	in	medicine	,	I	think	,	are	baffled
Ref	Je	pense	que	en	médecine	nous	sommes	dépassés	
NMT	Nous	,	en	médecine	,	je	pense	,	sont	UNK
NMT+UR	Nous	,	en	médecine	,	je	pense	,	sont	sidérée
BPE	nous	,	en	médecine	,	je	pense	,	sommes	bafés
Subwords	nous	,	en	médecine	,	je	pense	,	sommes	b@@ af@@ és
FNMT	nous	,	en	médecine	,	je	pense	,	sont	déconcertés
Lemmas	lui	,	en	médecine	,	je	penser	,	être	déconcerter
Factors	pro-1-p-l	pct-l	prep-l	nc-f-s-l	pct-l	cln-1-s-l	v-P-1-s-l	pct-l	v-P-3-p-l	vppart-K-m-p-l

Table 4.14: Examples of translations with NMT, NMT applying unknown replacement (UR) method, BPE and FNMT (without unknown words replacement) systems. The dataset used is IWSLT'15.

Table 4.14 shows another example comparing NMT, BPE and FNMT systems and the unknown replacement method proposes "sidérée" with an incorrect number. The NMT system generated an unknown token (UNK) when translating the English word *"baffled"*. We observe that BPE translates *"baffled"* to *"bafés"* which does not exist in French. This error probably comes from the shared vocabulary between the source and target languages creating an incorrect word very similar to its aligned source tokens. FNMT translates it to *"déconcertés"* which is a better translation than in the reference. One should note that it is not generated by the unknown word replacement method. However, for this particular example, an error on the factors leads to the word *"sont"* instead of *"sommes"*, resulting in lower automatic scores for FNMT output.

4.5.3 Factors applied in the source language

Recently, linguistic input features have improved NMT (Sennrich and Haddow, 2016). Extending the approach with input factors could make the target language factors generation easier having their corresponding lemmas and factors at the source side as well. Translating from factors to factors seems to be more sensible and natural.

Furthermore, it can also allow generating unseen words in the training data and managing more vocabulary including all the derivative forms of the lemmas in the source language. Therefore, we have extended the architecture shown in Chapter 2, Section 2.2 to support several input sequences.

The **encoder** supporting both input sequences is the combination of two encoders, the encoding of the lemmas and factors of the source language. The anno-

tations of the sequences in a forward and backward representation are presented in Equation 4.21 for lemmas and Equation 4.22 for factors.

$$\mathbf{a}_i^L = \begin{bmatrix} \overrightarrow{h_i}^L \\ \overleftarrow{h_i}^L \end{bmatrix} \tag{4.21}$$

$$\mathbf{a}_i^F = \begin{bmatrix} \overrightarrow{h_i}^F \\ \overleftarrow{h_i}^F \end{bmatrix} \tag{4.22}$$

We have combined both annotations in different manners to be able to train the model.

- The first model we have trained (**sum** model) combines both annotations summing them to have only one annotation (see Equation 4.23). This is possible because both sequences always have the same length and they are related to each other. The rest of the model (attention mechanism and decoder) remains the same.

$$\mathbf{a}_i = \mathbf{a}_i^L + \mathbf{a}_i^F \tag{4.23}$$

Inspired by Caglayan et al. (2016), we have trained several other models with different settings of attention mechanism. Those models allow the model to share or separate the attention mechanism weights between the two input sequences. These settings consist of sharing or using different transformation weights for the input context of the encoder or the hidden state of the decoder in the attention mechanism. The lemmas and factors at the output side keep sharing the same hidden states.

The morphological agreement between the words in the sentence is a monolingual task but the information of which factors should be produced in target can be given from source factors. In the example shown in Table 4.15, we observe the factors in the source and target languages. In order to be able to choose the correct inflected form of "soutenir" ("defend" in English) in the target language, we need to know the person and number in the source language. This information is given in the first factors of the source sentence (1st person and singular in the example). Target factors production needs to know source factors but this is not needed for target lemma production (only second lemma of the source sentence is necessary). This is the motivation of using different attention mechanism for lemmas and factors.

As we explained in Equation 2.28, the attention weights of each annotation are computed using a feed-forward network.

- The first setting consists of sharing all the weights in the attention mechanism. The attentional weights of the lemma input sequence are defined with e_{ij}^L and those for the factors input sequence with e_{ij}^F.

	Source language			
word/lemma	I	defend	my	thesis
factors	pronoun 1st person singular	verb present	pronoun	noun
	Target language			
word	Je	soutiens	ma	thèse
lemma	Je	soutenir	ma	thèse
factors	pronoun 1st person singular	verb 1st person singular present	pronoun feminine singular	noun feminine singular

Table 4.15: Example using factors in source and target languages.

$$e_{ij}^{L} = \mathbf{v_a} \tanh(\mathbf{W}_D \mathbf{s}'_j + \mathbf{W}_E \mathbf{a}^L) \tag{4.24}$$
$$e_{ij}^{F} = \mathbf{v_a} \tanh(\mathbf{W}_D \mathbf{s}'_j + \mathbf{W}_E \mathbf{a}^F) \tag{4.25}$$

where $\mathbf{W}_D$ are the transformation weights of the hidden state of the decoder and $\mathbf{W}_E$ the transformation weights of the input context of the encoder. $\mathbf{v_a}$ is another transformation of the attention mechanism. $\mathbf{s}'_j$ is defined previously in Equation 2.14.

The weights of the attention mechanism are computed as usual. e_{ij}^{L} receives the lemmas annotations ($\mathbf{a}^L$) and e_{ij}^{F} receives the factors annotations ($\mathbf{a}^F$). In this setting, both encoder and decoder weights matrices are shared by the two input sequences being independent on the input sequence (**ind-ind** model).

- The second option includes distinct encoder weights matrices and same decoder transformation weights in the attention mechanism. vv_a are also distinct for lemma and factors input sequence.

$$e_{ij}^{L} = \mathbf{v}_{\mathbf{a}}^{L} \tanh(\mathbf{W}_D \mathbf{s}'_j + \mathbf{W}_E^{L} \mathbf{a}^L) \tag{4.26}$$
$$e_{ij}^{F} = \mathbf{v}_{\mathbf{a}}^{F} \tanh(\mathbf{W}_D \mathbf{s}'_j + \mathbf{W}_E^{F} \mathbf{a}^F) \tag{4.27}$$

where e_{ij}^{L} receives a specialized weight matrix $\mathbf{W}_E^L$ and specialized transformation $\mathbf{v}_{\mathbf{a}}^{L}$ for lemmas. e_{ij}^{F} receives $\mathbf{W}_E^F$ and $\mathbf{v}_{\mathbf{a}}^{F}$ for factors. The input context transformation weights of the encoder is dependent on each input sequence (both are distinct) and $\mathbf{W}_D$ is shared by the two input sequences (**dep-ind** model).

- The third setting is the opposite as previous one.

$$e_{ij}^{L} = \mathbf{v_a} \tanh(\mathbf{W}_D^L \mathbf{s}'_j + \mathbf{W}_E \mathbf{a}^L) \tag{4.28}$$
$$e_{ij}^{F} = \mathbf{v_a} \tanh(\mathbf{W}_D^F \mathbf{s}'_j + \mathbf{W}_E \mathbf{a}^F) \tag{4.29}$$

$\mathbf{W}_E$ is shared and $\mathbf{W}_D^L$ is a specialized matrix for lemmas input sequence and $\mathbf{W}_D^F$ for factors input sequence. $\mathbf{W}_E$ and $\mathbf{v_a}$ are independent (shared) and the hidden state transformation weights of the decoder is dependent (distinct) on the input sequence (**ind-dep** model).

- The last setting consists of having both weights matrices distinct. This setting has two totally separated attention mechanisms.

$$e_{ij}^{L} = \mathbf{v}_\mathbf{a}^L \tanh(\mathbf{W}_D^L \mathbf{s}'_j + \mathbf{W}_E^L \mathbf{a}^L) \tag{4.30}$$
$$e_{ij}^{F} = \mathbf{v}_\mathbf{a}^F \tanh(\mathbf{W}_D^F \mathbf{s}'_j + \mathbf{W}_E^F \mathbf{a}^F) \tag{4.31}$$

The transformation weights of the encoder and the transformation weights of the decoder are dependent on the input sequences (**dep-dep** model).

The use of specialized matrices increases the number of parameters.

The **decoder** has been modified to combine both annotations. For that purpose, GRU_1 is extended to integrate multiple input sequences. GRU_1 initialization is done with the sum of the two annotations.

$$\mathbf{s}'_0 = \tanh\left(\mathbf{W}_{init}\left(\frac{1}{N}\sum \mathbf{a}_i^L + \mathbf{a}_i^F\right) + \mathbf{b}_{init}\right) \tag{4.32}$$

At each timestep, this special attention mechanism generates two specific context vectors ($\mathbf{c}_j^L$ and $\mathbf{c}_j^F$).

$$\mathbf{c}_j^L = \sum_{i=1}^{N} \alpha_{ij}^L \mathbf{a}_i^L \tag{4.33}$$
$$\mathbf{c}_j^F = \sum_{i=1}^{N} \alpha_{ij}^F \mathbf{a}_i^F \tag{4.34}$$

The global context is then obtained by applying another tanh operation over the concatenation of two contexts to be able the fusion of them.

$$\mathbf{c}_j = \tanh\left(\mathbf{W}_{fusion}\begin{bmatrix}\mathbf{c}_j^L \\ \mathbf{c}_j^F\end{bmatrix} + \mathbf{b}_{fusion}\right) \tag{4.35}$$

Then, GRU_2 receives the global context $\mathbf{c}_j$ and the output of GRU_1.

$$\mathbf{s}_j = \text{GRU}_2\left(\mathbf{s}'_j, \mathbf{c}_j\right) \tag{4.36}$$

Results

We have performed experiments combining the different word representations (word, BPE and Factors) at source and target sides. When applying factors at input side, we have used the models previously presented to combine the two input sequences. In this experiment, we also applied unknown words replacement when the output side corresponds to word or factors levels.

	METEOR↑	BLEU↑		
Model	**word**	**word**	**lemma**	**factors**
Words-Words / +UR	62.21 / 63.38	41.80 / 42.74	45.10	51.80
BPE-BPE	62.87	42.37	45.96	53.31
Words-Factors / +UR	**64.10 / 64.98**	**43.42 / 44.15**	47.18	54.24
Factors-Words sum/ +UR	61.83 / 62.97	41.24 / 42.25	44.76	51.53
Words-BPE	63.06	43.34	46.56	54.60
BPE-Factors / +UR	62.58 / 63.43	42.11 / 42.83	45.88	53.07
Factors-Factors sum/ +UR	63.95 / 64.66	42.99 / 43.63	47.00	54.02
Factors-Factors ind-ind/ +UR	63.95 / 64.71	42.84 / 43.48	46.98	54.19
Factors-Factors ind-dep/ +UR	62.29 / 62.97	41.79 / 42.35	45.85	52.33
Factors-Factors dep-ind/ +UR	62.16 / 62.83	41.56 / 42.13	45.82	52.24
Factors-Factors dep-dep/ +UR	63.72 / 64.40	43.34 / 43.95	**47.30**	**54.84**

Table 4.16: Results on IWSLT test15 comparing factors, words and BPE approach at input and output side of the network. BLEU and METEOR performance of NMT and FNMT systems with and without `UNK` replacement (UR) are presented.

Table 4.16 presents the results. The first three rows have been already presented in the previous experiment. They correspond to the words-to-words NMT model, BPE-to-BPE model and Words-to-Factors FNMT model.

We first combine factors at the input side and words at the output side (*Factors-Words* model) using the sum of the two annotations. This model obtained the worst result. The source language is English which is grammatically simple and factors only on the source side do not give any improvement.

We applied BPE only at the output side (*Words-BPE* model). This model does not improve the *BPE-BPE* model at word and lemma levels but it obtains a good score at factors level compared to the rest of the models. BPE at output side helps the prediction of factors.

In order to create a model with the benefits of BPE and FNMT systems, we combined both systems applying BPE at input side and factors at output side (*BPE-Factors* model). Unfortunately, this model did not improve the others, it only improves the *Words-Words* model. Applying BPE only at the source side does not give the best benefit to the model.

The different combinations of the two input sequences have been experimented in different models. First, the model with the sum of the two annotations gives a competitive score approaching the best model score (-0.15 in BLEU points and -0.43

in METEOR points without UNK replacement). The *ind-ind* model which shares the attention mechanism matrices has a similar behaviour. The *ind-dep* and *dep-ind* models get more than one BLEU and METEOR point than *ind-ind* model and *dep-dep* model. The idea of mixing the weights in a shared or separated manner is not good for the factored approach because lemmas and factors are related to each other. The last model is *dep-dep*, which has separated weights matrices for each input sequence. This model did not improve either the *Words-Factors* model at word level, but the scores at lemma and factors levels are the best of all presented models. This means that separating the lemma and factors weights matrices helps the generation of lemmas and factors which seems to be coherent. On the contrary, *dep-dep* model does not improve at word level. *dep-dep* model has more specialized weights for each input sequence and more shared weights are needed in order to find good surface forms combining the generated lemmas and factors.

The UNK replacement method improves about 1 BLEU and METEOR point using words at output side. When applying factors at output side the improvement is smaller, between 0.60 and 0.85 points. This is because the FNMT system does not generate as much UNK as NMT system using words at output side.

In conclusion, factors applied in the source language do not give any improvement in the translation in these experiments.

According to the conclusion in Belinkov et al. (2017), the lower layers are better at capturing morphology, while higher ones are focused on the meaning, improving the translation performance. Moreover, the results in Dalvi et al. (2017) show that the decoder learns considerably less amount of morphology than the encoder and the overall system does not learn as much about target morphology as source morphology. Therefore, the input side of the network is already good at capturing morphology so the factored level in the source language does not improve the translation. On the other hand, the factored level at output side helps to obtain better translation.

Additionally, the task of this work is English to French translation, English is a morphologically simpler language than French and factored approach does not help for its translation. Further experiments can be performed to check if the factors at input side can benefit morphologically rich languages. In the rest of experiments, we apply the factors only at the target side.

4.6 Conclusion

In this chapter, a new approach based on linguistic factors has been introduced. This approach is called Factored Neural Machine Translation (FNMT). The motivation (Section 4.1) and description (Section 4.2) of the FNMT approach have been explained in the first and second section, respectively.

Section 4.3 presents a set of preliminary experiments in order to test the FNMT models. We have trained different FNMT models reducing the output layer size showing the generalization power of the approach. We compared the FNMT system

to other state of the art systems as a word level NMT system, a BPE system or a multilingual system which can generate several sequences. Better results have been obtained with FNMT system than the rest of the systems.

In Section 4.4, several options of FNMT system have been experimented. Dependency models introducing additional links to the outputs have been implemented and evaluated. Improvements of the FNMT system in the automatic metrics have not been shown using dependency models. Different options of feedback in the model have been tested using only the previous generated lemma or factors, or a combination of both of them. We saw that the use of the previous generated lemma as feedback performs better because the main information of the word is contained in the lemma. The concatenation of the two previous generated lemmas and factors usage is a good practice in the FNMT system because more information is included in the model.

The last set of experiments of this chapter is presented in Section 4.5. Case sensitive data have been used in order to have a more realistic task and increase the difficulty of the translation process having more vocabulary. The FNMT system gain over the rest of the systems is augmented compared to the lowercased data results. A qualitative analysis has been performed observing the alignments from the attention mechanism and showing some translation examples of the systems. The translation outputs show the advantages of the FNMT system producing less `UNK` tokens than NMT system and always generating correct words compared to BPE systems. Factored approach has been also applied at the input side of the model (Section 4.5.3). We did not observe an improvement using factors in the source language. This can be due to the grammatically simple language used which has been English. Further experiments can be performed to check if the factors at input side can benefit morphologically rich languages. Moreover, we can exchange the translation direction to verify the usefulness of the factors in source language.

4.7 Related publications

The related publications of this chapter are:

> García-Martínez, M., Barrault, L., and Bougares, F. (2016). Factored Neural Machine Translation Architectures. In *Proceedings of the International Workshop on Spoken Language Translation*, IWSLT'16, Seattle, USA

> García-Martínez, M., Barrault, L., and Bougares, F. (2017). "Neural Machine Translation by Generating Multiple Linguistic Factors". In Camelin, N., Estève, Y., and Martín-Vide, C., editors, *Statistical Language and Speech Processing*, pages 21–31, Cham. Springer International Publishing

> Caglayan, O., García-Martínez, M., Bardet, A., Aransa, W., Bougares, F., and Barrault, L. (2017). NMTPY: A Flexible Toolkit for Advanced Neural Machine Translation Systems. *Prague Bull. Math. Linguistics*, 109:15–28

CHAPTER 5

MORPHOLOGICALLY RICH LANGUAGES TRANSLATION

Contents

"I just feel so alone, even when I'm surrounded by other people." Charlotte (Lost in translation)

Translation into a morphologically rich language requires a large output vocabulary to model various morphological phenomena. Open vocabularies remain a challenge for Neural Machine Translation (NMT) (Cho et al., 2014a; Bahdanau et al., 2014), both for learnability and computational reasons. Morphological variation and lexical productivity cause word forms unseen in training. Increasing the

vocabularies partially mitigates these issues but we face both previously mention issues (see Section 1.2): (1) data sparsity due to the difficulty of modeling rare seen or unseen inflected forms and (2) a larger target vocabulary increases the computational complexity of the output layer. Factored systems could mitigate those issues because they can cover more vocabulary.

In this chapter, we test the FNMT system translating in other more difficult tasks to see its advantages. Firstly, we translate from English into two highly inflected languages: Czech and Latvian. For these two language pairs, the training dataset is large. Secondly, we translate Arabic which is a morphologically rich language into French. We experimented with this language pair under low resource conditions and compare it to a scenario with more resources. Lastly, we provide human evaluation results and we show an analysis of the translation outputs.

5.1 High resource conditions

The language pairs English-to-Czech and English-to-Latvian are translated using a Factored NMT (FNMT) system where two symbols are generated at the same time (see Section 4.2 for more details). The FNMT systems are compared to a baseline NMT system. This work was performed in the WMT'17[1] evaluation campaign.

5.1.1 Model architectures

Our baseline NMT system is the same as presented in Chapter 2, Section 2.2 adding some improvements. Firstly, we use tied embeddings as in Chapter 4, Section 4.5. Secondly, we equipped the encoder with layer normalization (Ba et al., 2016), a technique which adaptively normalizes the incoming activations of each hidden unit with a learnable gain and bias.

We experimented with two types of FNMT systems which have a second output in contrast to baseline NMT system. Their architectures differ after the generation of the decoder state. The first one contains a single hidden-to-output layer (*h2o*) which is defined in Equation 5.1.

$$\phi\left(\sum(\mathbf{s}_{j-1}, \mathbf{c}_j, [\mathbf{E}[y^L_{t-1}]; \mathbf{E}[y^F_{t-1}]] \cdot W_{fb})\right) \tag{5.1}$$

where $\mathbf{s}_{j-1}$ is the hidden state, $\mathbf{c}_j$ is the context vector, $\mathbf{E}[y^L_{t-1}]$ is the lemma embedding, $\mathbf{E}[y^F_{t-1}]$ is the factors embedding and W_{fb} is a weight matrix of the feedback.

The single *h2o* layer is then used by two separated softmax layers, this is the same model as previous experiments (see Section 4.2, Figure 4.2 on page 43).

The second system contains two separated *h2o* layers, each one specialized for a particular output (see Figure 5.1).

Equation 5.2 defines the specialized *h2o* layer for lemmas and Equation 5.3 defines the specialized *h2o* layer for factors.

[1] WMT'17 website: http://www.statmt.org/wmt17

$$\phi\left(\sum(\mathbf{s}_{j-1}, \mathbf{c}_j, \mathbf{E}[y^L_{t-1}])\right) \tag{5.2}$$

$$\phi\left(\sum(\mathbf{s}_{j-1}, \mathbf{c}_j, \mathbf{E}[y^F_{t-1}])\right) \tag{5.3}$$

where $\mathbf{s}_{j-1}$ is the hidden state, $\mathbf{c}_j$ is the context vector, $\mathbf{E}[y^L_{t-1}]$ is the lemma embedding and $\mathbf{E}[y^F_{t-1}]$ is the factors embedding.

The decoder feedback is modified to use information from the multiple outputs. The concatenation of the embeddings of the pair of generated symbols is used to feed the decoder's cGRU at each timestep. This layer uses the context vector, the decoder's hidden state and the concatenation of the embeddings of the previous generated tokens.

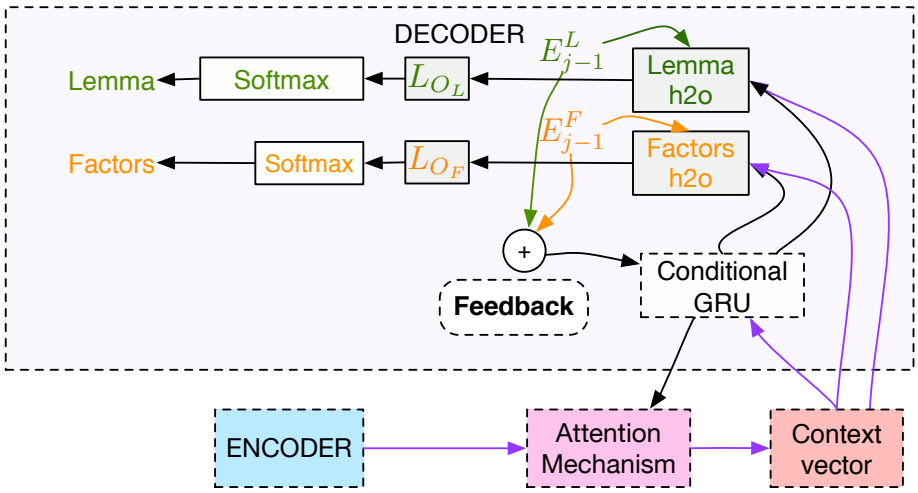

Figure 5.1: Factored NMT system with separated *h2o* layers.

Both FNMT systems are similar excepting in that the first model (single *h2o* layer) uses the concatenation of the embeddings of the previously generated factors (see Equation 5.1), and the second model (separated *h2o*) uses one *h2o* layer for each output receiving only the embedding of the symbol is generating (see Equation 5.2 for lemmas and Equation 5.3 for factors). The two separated *h2o* layers allow the system to have more specialized weights for each output (see Figure 5.1).

5.1.2 Experimental framework

We introduce here the experimental setup for all the reported systems translating from English into Czech and Latvian. Our experimental setting follows the guidelines of the WMT'17[2] news translation task.

[2] http://www.statmt.org/wmt17/translation-task.html

Data and preprocessing

All datasets are tokenized and truecased using the Moses toolkit (Koehn et al., 2007). PoS tagging in Czech data is performed with Morphodita toolkit (Straková et al., 2014). All the Latvian preprocessing was provided by TILDE[3]. This preprocessing consists of using the TILDE's regular expression-based tokeniser that takes into account how the languages express abbreviations, contractions, date, time, numerical expressions, etc. and non-translatable entities like phone numbers, e-mail addresses, URLs, codes, etc. Only the first word in each sentence is truecased. Latvian PoS-tagging is done with the LU MII Tagger (Paikens et al., 2013). After preprocessing, we filter out training sentences with a maximum length of 50 or with a source/target length ratio higher than 3.

The English-to-Czech systems are trained using approximately 20M sentences from the relevant news domain parallel data: news-commentary, Europarl and specific categories of the *Czeng* corpus (news, paraweb, EU, fiction) (see Table 5.1 for more details). Early stopping is performed using newstest2015, newstest2016 is used as internal test set and newstest2017 as official test set.

Data	Name	# Sents	# Words EN / CS
Training	All bitext	20M	433M / 394M
	news-commentary	211k	5.1M / 4.7M
	Europarl	642k	17M / 15M
	Czeng news	249k	6M / 5M
	Czeng paraweb	2M	38M / 34M
	Czeng EU	10M	281M / 256M
	Czeng fiction	6M	86M / 79M
Training	All synthetic	14M	322M / 292M
	news-2016	6M	103M / 93M
	Czeng EU subset	5M	147M / 133M
Development	newstest2015	2.6k	53k / 46k
Test	newstest2016	3k	64k / 56k
Official test	newstest2017	3K	61k / 55k

Table 5.1: WMT'17 data for English-to-Czech translation.

Synthetic data is generated from *news-2016* monolingual corpus provided by Sennrich et al. (2016b). The Czech monolingual corpus *news-2016* was backtranslated to English using the single best system provided by the University of Edinburgh from WMT'16[4].

In order to focus more on the in-domain data, five copies of news-commentary[5],

[3] www.tilde.com

[4] http://data.statmt.org/rsennrich/wmt16_systems/

[5] Adding multiple copies of the same corpus into the training set can be seen as a coarse way to weight different corpora and favor in-domain bibtext.

the news subcorpus from *Czeng*, as well as 5M sentences from the *Czeng* EU corpus randomly selected after running modified Moore-Lewis filtering with XenC (Rousseau, 2013) are added to the backtranslated data. We end up with about 14M sentences and 322M words for English and 292M for Czech.

The English-to-Latvian systems are trained using all the parallel data available for the WMT17 evaluation campaign. Using the Microsoft sentence aligner[6], sentences which are not aligned one-by-one were filtered out in The Digital Corpus of the European Parliament (DCEP). Also, modified Moore-Lewis filtering was used to select sentences in DCEP corpus. We kept the best 1M sentences, which led to a total of almost 2M parallel sentences. The validation set consists of 2k sentences extracted from the Latvian information agency (LETA) corpus, newsdev2017 is used as internal test set and newstest2017 is the official test set.

Training was carried out for a part of these systems on synthetic parallel data. Monolingual corpora *news-2015* and *news-2016* were backtranslated with a Moses system (Koehn et al., 2007). Similarly to Czech, we added ten copies of the LETA corpus which contains only 14k sentences and two copies of Europarl (637k sentences) and *rapid* (306k sentences) which are bigger corpora performing corpus weighting. The final corpus contains 7M sentences and 172M words for English and 143M words for Latvian. Table 5.2 shows a summary of the quantity of available data.

Data	Name	# Sents	# Words EN / LV
Training	All bitext	2M	50M / 42M
	DCEP filtered	1M	26M / 22M
	rapid	306k	6M / 5M
	farewell	10k	105k / 558k
	Europarl	637k	17M / 14M
Training	All synthetic	7M	172M / 143M
	news-2015	3M	68M / 57M
	news-2016	2M	54M / 45M
	LETA	14k	443k / 371k
Development	LETA subset	2k	68k / 58k
Test	newsdev2017	2k	49k / 41k
Official test	newstest2017	2k	47k / 39k

Table 5.2: WMT'17 data for English-to-Latvian translation.

System setup

For the training of all models, we used `NMTpy` toolkit. All models use **512**-dimensional embeddings and GRU layers with **1024** hidden units for both the encoder and the

[6] https://www.microsoft.com/en-us/download/details.aspx?id=52608

decoder. Dropout is enabled on source embeddings, context vector, as well as output layers. When training starts, all parameters are initialized with Xavier (Glorot and Bengio, 2010a). In order to slightly speed up the training on the actual parallel data, the learning rate was set to 0.0004, patience to 30 with a validation every 20k updates.

We also trained systems with synthetic data which are initialized with a previously trained model on the provided bitext only. For these systems, the learning rate is set to 0.0001 and the validations are performed every 5k updates in order to avoid overfitting on synthetic data and forgetting the previously learned weights. These systems were tuned with Adam optimizer (Kingma and Ba, 2014). Two models with different seeds are trained for each system for ensembling purposes. Ensembling several models with the same architecture during decoding allows the system to use the scores of all the ensembled models. This method improves the final translation performance (Sennrich et al., 2016a). The hardware used for training the models is the GPU Tesla K40c. The memory taken for the process is 11MB. Training each model takes one month.

The vocabulary size used for source language at word level and target language at word and lemma level is 90k. For FNMT models, the vocabulary size of the second output for factors is only 1.5k for Czech and 376 for Latvian. The number of parameters in FNMT systems increases around 2.2% for Czech and 1.7% in Latvian. Bilingual BPE models (BPE dictionary built with source and target language, see Section 3.1.6 for more details) for each language pair and system setup were learned on the parallel data. 90k merge operations were performed. The sentences longer than 50 tokens are filtered out after BPE segmentation. For FNMT systems, BPE is applied on the lemma sequence and the corresponding factors are repeated when a split occurs. We call this system FBPE.

In order to compare FNMT and FBPE systems with other existing factored systems using lemmas and factors, we interleaved the lemmas and factors in a single sequence, as done in Tamchyna et al. (2017). We called this model Interleaved FNMT (IFNMT model). In Section 3.2.3, the reader can find more details about this work. Table 5.3 shows how the outputs are represented in the three systems using factors.

Model	1st output	2nd output
FNMT	ressembler	v-C-3-p
FBPE	ressembl+ er	v-C-3-p v-C-3-p
IFNMT (Tamchyna et al., 2017)	ressembler v-C-3-p	-

Table 5.3: Factored representations at target side of the English to Czech/Latvian translation systems. IFNMT system is only used for English to Czech translation.

FNMT postprocessing: from factors to words

The factored systems predict at each timestep a lemma and its factors (PoS-tag, gender, number, etc.) which requires a non-trivial additional step producing sentences in a fully inflected language.

Given the lemma and the factors, word forms are retrieved with a dictionary look-up. In the context of morphologically rich languages, deterministic mappings from a lemma and its factors to a surface form are very rare. Instead, the dictionary often contains several word forms corresponding to the same lemma and morphological analysis.

A first way to solve this ambiguity is to simply compute unigram frequencies of each word form, which was done over all the monolingual data available at WMT'17 for both Czech and Latvian. During a dictionary look-up, ambiguities can then be solved by taking the most frequent word form.

For Czech reinflection, we used the Morphodita generator (Straková et al., 2014). Since there is no tool for Latvian to convert factors to words, all the available WMT'17 monolingual data has been automatically tagged using the LU MII Tagger (Paikens et al., 2013) and the result is kept in a look-up table. This look-up table maps the lemmas and factors to their corresponding word. As one could expect, we obtained a large table (nearly 2.5M forms) in which we observed a lot of noise.

Reranking

Taking only the most frequent surface form from each {lemma, factors} pair, the target sequence context is ignored which can be important to generate a correct translation.

To avoid this, we used two types of hypothesis reranking using the best FNMT system. For each hypothesis, we generate the surface form with the *factors-to-word* postprocessing. Since a single {lemma, factors} pair may lead to multiple possible words, k possible words are considered for each pair. For Czech, $k = 10$, a low value because this language has a specialized tool for reinflection. On the other hand, the surface forms proposals for the same lemma and factors in Latvian are many. Therefore, we included 100 hypothesis for k-best reranking in Latvian in order to mitigate the poor quality of this dictionary by relying more on the reranking. Additionally to the k-best hypothesis, n-best hypothesis were taken from the beam search (n= 12). Finally, the hypotheses are reranked with the best BPE-based NMT model to select the 1-best hypothesis.

For English-to-Latvian, we performed n-best reranking for BPE and FBPE systems using the scores of two Recurrent Neural Network Language Models (RNNLM):

1. A simple RNNLM (Mikolov et al., 2010)

2. A GRU-based RNNLM included in `NMTpy`.

The RNNLMs are trained on WMT'17 Latvian monolingual corpus and the target side of the available bitext (175.2M words in total). For FBPE system, the

log probability obtained by our best BPE model is also used in addition to the RNNLM scores. The reranking is done using the *nbest* tool provided by the CSLM toolkit[7] (Schwenk, 2010). The score weights were optimized with CONDOR (Vanden Berghen and Bersini, 2005) to maximize the BLEU score on newsdev2017 set.

Constrained decoding

FNMT system predicts a lemma and its factors at each timestep. In the architecture, after sharing the same context vector and decoder state, both outputs are independent. As a consequence, the best lemma hypothesis can be incompatible with the best factors hypothesis, and it can get worse when n-best hypotheses are considered in beam search.

The constrained decoding procedure makes sure that the generated factors are compatible with the generated lemma at each timestep. This procedure consists of associating each lemma in the target vocabulary with the corresponding possible factors in a dictionary. For that, we built a lookup table with the lemmas in the vocabulary as keys and the possible factors for each lemma as values.

The decoding is modified as follows: for each candidate lemma, we only retain the compatible factors checking the lookup table, and select the n-best hypotheses to be kept in the beam from this filtered list. This constraint ensures that the beam search does not evaluate incompatible pairs of factors. (*e.g.* factors including adjective with lemma *book* which is a noun or a verb).

Creating such a mapping is trivial for completed lemmas, but less obvious when splitting in BPE units. Since a BPE unit can be generated from different lemmas having different factors, the size of the set of possible factors can grow quickly. Due to these reasons, we did not apply the constrained decoding in FBPE systems.

5.1.3 Automatic evaluation

The experiments are carried out on the data from the WMT'17 evaluation campaign as described previously. Firstly, we experimented with the NMT architecture at word level and also using interleaved FNMT representation (lemmas and factors in a single sequence). Secondly, the FNMT architecture (see Figure 5.1) was tested comparing the usage of a single or separated *h2o* layers. Each FNMT model is tested with and without constrained decoding. Additionally, NMT and FNMT models are processed with BPE units (BPE and FBPE systems). Finally, we report results adding the synthetic data and using reranking and ensembling techniques.

Results are reported using the following automatic metrics: BLEU (Papineni et al., 2002b), BEER (Stanojević and Sima'an, 2014) which tunes a large number of features (permutation trees (Zhang et al., 2008), adequacy, fluency, F-score, Kendall (Isozaki et al., 2010) and ordering) to maximize the human ranking correlation at sentence level and corpus level using a linear interpolation and CharacTER (Wang

[7] http://github.com/hschwenk/cslm-toolkit

et al., 2016), a character-level version of Translation Edit Rate (TER) which has shown a high correlation with human rankings (Bojar et al., 2016).

Experiments without synthetic data

The results using only the bitext provided at the WMT'17 the evaluation campaign are presented in Table 5.4 for English-to-Czech. At decoding time, Czech systems performed better with a beam size of 2, which was used to provide these results. Table 5.5 shows the results for English-to-Latvian. The BLEU scores reported are computed with *multi-bleu.perl* which makes them consistently lower than official evaluation matrix scores[8].

	newstest2016			**newstest2017**		
Model	BLEU ↑	BEER ↑	CTER ↓	BLEU ↑	BEER ↑	CTER ↓
NMT	15.74	47.50	74.43	12.54	44.96	78.68
IFNMT	15.47	48.67	71.98	12.99	46.36	74.38
FNMT						
single *h2o* layer	16.73	50.50	65.51	14.09	48.15	69.85
+ constrained dec.	17.42	50.94	64.95	14.93	48.76	69.26
sep. *h2o* layers	16.54	50.12	66.35	13.89	47.78	70.63
+ constrained dec.	17.56	50.73	65.48	14.66	48.26	69.96
BPE	**18.30**	52.18	**60.55**	**14.90**	49.35	65.68
FBPE						
single *h2o* layer	17.30	51.82	61.19	14.19	48.98	66.28
sep. *h2o* layers	17.34	**52.22**	60.62	14.73	**49.61**	**65.34**

Table 5.4: Scores for English-to-Czech systems trained on WMT'17 official bitext data

	newsdev2017			**newstest2017**		
Model	BLEU ↑	BEER ↑	CTER ↓	BLEU ↑	BEER ↑	CTER ↓
NMT	14.51	47.47	77.24	9.90	42.64	86.04
FNMT						
single h2o layer	13.96	49.53	68.36	9.68	45.24	77.07
+ constrained dec.	14.02	49.48	69.97	9.94	45.21	78.11
sep. h2o layers	13.92	49.93	68.45	9.71	45.10	77.51
+ constrained dec.	14.38	49.74	70.04	10.07	45.26	78.08
BPE	**15.25**	50.61	**64.56**	10.36	46.23	**72.13**
FBPE						
single h2o layer	14.45	**50.86**	67.14	10.45	46.36	72.25
sep. h2o layers	14.39	50.72	66.05	**10.69**	**46.44**	72.96

Table 5.5: Scores for English-to-Latvian systems trained on WMT'17 official bitext.

[8]MT evaluation matrix: http://matrix.statmt.org

The English-to-Czech **IFNMT** system (see Table 5.3 for details) results can be seen in Table 5.4. IFNMT system performs worse than BPE, FNMT and FBPE systems and only better than NMT system when translating newstest2017. FNMT system has only few parameters more than IFNMT system because FNMT system incorporates the small second output layer. IFNMT system does not have problems of postprocessing dealing with the interleaved sequence: both predictions contain the same number of lemmas and factors and there are no lemmas in the positions of factors or vice versa. Indeed, the lemmas and factors generated by IFNMT system are worse than FNMT systems. As a consequence, we can confirm that FNMT architecture can learn better predictions of lemmas and factors. There are two reasons of this, (1) FNMT system have specialized weights for lemmas and factors and (2) FNMT generates lemmas and factors synchronously.

We can observe that using the **constrained decoding** (see Section 5.1.2) consistently improves the results. Constrained decoding has not been used with FBPE system (last two rows of Table 5.4) because a BPE token may correspond to a large set of factors. Consequently, it is very unlikely that such a method would provide improvement, since it would not constrain the factors output much. In the Latvian setup, we observe a small improvement in terms of BLEU using constrained decoding (see Table 5.5). In some cases, when translating newsdev2017 and newstest2017 for single *h2o* layer FNMT system, BEER and CTER evaluation obtain even worse scores using constrained decoding. When translating in Czech, the constrained decoding gives a bigger improvement than when applied in Latvian (see Table 5.4). This is probably due to the poor quality of the look-up table we have created for the Latvian reinflection (see Section 5.1.2 for more details).

Two FNMT models were tested, one model using a **single *h2o* layer** and the other model using **two separated *h2o* layers**. We observe mixed results, the single *h2o* layer FNMT system has slightly better scores than separated *h2o* layers FNMT system, except for English-to-Latvian newstest2017 translation where the opposite occurs. On the other hand, FBPE system obtains better performance using separated *h2o* layers with the only exception of English-to-Latvian newsdev2017 evaluation where FNMT system using single *h2o* layer FNMT obtains better scores than using separated *h2o* layers. As a consequence, we decided to only use the separated *h2o* layers model for the next set of experiments including synthetic data.

For both language pairs, the systems using **BPE** tokens significantly outperform word-level NMT systems. The results show that BPE units applied in factors (FBPE model) are always better than FNMT model when constrained decoding is not applied. When adding constrained decoding to FNMT system, FBPE is better than FNMT in BEER and CTER evaluation metrics. For English-to-Czech translation, FBPE models BLEU scores are slightly lower than FNMT with constrained decoding in newstest2016. In newstest2017, single *h2o* layer option obtains better scores when is applied in the FNMT system with constrained decoding. On the contrary, separated *h2o* layers option is better when is applied in FBPE system. For English-to-Latvian newsdev2017 translation, FBPE model BLEU scores are better than FNMT when using single *h2o* layer and similar when using separated *h2o* layers.

When evaluating BLEU with newstest2017, FBPE performs better than FNMT even applying constrained decoding. Globally, when applying BPE method, including FBPE, the systems perform better.

Finally, we compare BPE and **FBPE** systems. The system behave different depending on the automatic metric. BLEU evaluation shows that BPE system performs better than FBPE system in all test sets except for English-to-Latvian newstest2017 where FBPE obtains the best result. On the other hand, when looking at the BEER scores, FBPE system always performs better than BPE system. The last metric (CTER) reflects different behaviour depending on the test dataset, BPE system performs better in all the test datasets except for English-to-Czech newstest2017 evaluation where FBPE system obtains the best result.

Due to these results, BPE method is always applied in the next experiments including synthetic data. Word-level NMT system and FNMT without BPE are no longer applied for this data. We also chose to only use separated *h2o* layers for FBPE models.

Experiments with synthetic data

In this set of experiments, we add the synthetic data and apply ensemble of models and reranking to boost the results. We compare again BPE and FBPE systems in order to analyse further their performances.

Tables 5.6 and 5.7 show the results of using selected parts of bitext and synthetic parallel data (see Section 5.1.2) for both language pairs. Each model trained with the selected bitext and synthetic data was initialized with the parameters of its counterpart trained on bitext only. The BPE vocabulary used was the same as in the model used for initialization, which can be led the systems to generate few unknown words. For example, we found 0.18% of `UNK` when translating newstest2017 into Czech with BPE system. In order to avoid `UNK`, we forced the decoder to avoid unknown token generation. When the `UNK` token obtains the highest score to be predicted, the system outputs the symbol with the second highest score, instead of `UNK`.

We also include results for setups that address the surface forms ambiguities issue when the system proposes several options for the same lemma and factors (see Section 5.1.2). The k-best setup performs the reinflexion of the 1-best sentences produced by the FNMT system and uses the ambiguities to generate k reinflected hypotheses out of one sentence (*+k-best reranking* in the tables). Additionally, the same procedure can be applied to the n-best hypotheses generated by the beam search procedure (in our case the beam size is empirically set to 12). The resulting nk-best hypotheses are in turn reranked with the BPE model in order to choose the surface form instead of relying on simple unigram frequencies.

Those results are referred to as *+n-best reranking* in the tables. Additionally, for English-to-Latvian systems, a reranking has been performed using two RNNLM.

Finally, for all the systems, we provide the results obtained with an ensemble of two systems with a different seed including the nk-best reranking procedure and

adding LM reranking for Latvian as final result.

	newstest2016			newstest2017		
Model	BLEU ↑	BEER ↑	CTER ↓	BLEU ↑	BEER ↑	CTER ↓
BPE	24.18	57.47	52.42	20.26	54.67	58.15
+ ensemble	**24.52**	**57.64**	**52.13**	**20.44**	54.77	58.04
FBPE						
sep. h2o layers	22.30	56.63	53.46	19.34	54.16	58.76
+ k-best reranking	22.35	56.60	53.49	19.36	54.17	58.71
+ n-best reranking	23.39	57.25	52.73	19.83	54.57	58.35
+ ensemble	24.05	57.59	52.27	20.22	**54.80**	**57.89**

Table 5.6: Scores for English-to-Czech systems trained on WMT'17 selected bitext and synthetic parallel data.

	newsdev2017			newstest2017		
Model	BLEU ↑	BEER ↑	CTER ↓	BLEU ↑	BEER ↑	CTER ↓
BPE	21.88	57.61	52.52	15.26	52.10	62.80
+LM reranking	21.98	57.68	53.07	15.59	52.27	63.05
+ensemble(no LM reranking)	22.34	57.99	**52.18**	15.46	52.40	62.44
+ensemble LM reranking	**22.46**	**58.02**	52.77	**16.04**	**52.66**	**62.40**
FBPE						
sep. h2o layers	18.93	56.01	54.36	13.98	51.26	63.90
+k-best reranking	20.56	56.94	53.42	14.80	51.78	63.19
+n-best reranking	21.59	57.62	52.83	15.31	52.34	62.64
+LM reranking	21.79	57.77	52.52	15.51	52.34	62.65
+ensemble(no LM reranking)	21.90	57.83	52.38	15.35	52.31	62.46
+ensemble LM reranking	21.87	57.74	53.04	15.53	52.40	62.80

Table 5.7: Scores for English-to-Latvian systems trained on WMT'17 selected bitext and synthetic parallel data.

We observe that including the synthetic parallel data, in addition to the provided bitext, results in a big improvement (generally +5 BLEU points) in BPE and FBPE systems for both language pairs (comparing Tables 5.4 and 5.6 for English-to-Czech and Tables 5.5 and 5.7 for English-to-Latvian).

k-best reranking of FBPE systems shows bigger improvement when translating into Latvian than into Czech. For Czech, the results are stable, whereas it provides an improvement of about +1 points for Latvian. This is due to the quality of the dictionary used for reinflection in each language. The Morphodita tool for Czech includes only good candidates. Therefore, a unigram model is sufficient to correctly transform the factors to words in Czech language. On the other hand, a similar tool is not available for Latvian, leaving more room for improvement.

n-best reranking always brings improvements in both language pairs and obtains similar performance to BPE systems.

For translation into Latvian, the reranking using RNNLMs gives an improvement of about +0.2 BLEU points for both BPE and FBPE systems. Applying the ensemble of several models also gives improvement for all systems. Finally, we applied RNNLMs reranking to the ensembled output for Latvian and obtained the best results. The BPE system obtains better scores than FBPE system when translating English-to-Czech newstest2016. When translating English-to-Czech newstest2017, BPE system has a small improvement of BLEU compared to the FBPE system. However, BPE system is similar in BEER and slightly higher in CTER than FBPE system. BEER and CTER automatic evaluation metrics are more correlated with human rankings than BLEU. This can give us and intuition that FBPE systems are more correlated with human translation than BPE systems.

The paper (Burlot*, F. and García-Martínez*, M. et al., 2017) shows additional experiments using normalised words (simplifying inflections as in English) and clustering of words representation performed by LIMSI research laboratory using FNMT architecture. The results for these experiments are better for FBPE system compared to BPE system.

We have compared FNMT systems with a very strong baseline including new state of the art techniques to push the results (BPE, reranking, ensembling and backtranslation), in order to observe how our systems in competitive conditions. This may harm showing the benefits of our systems because it is very hard to beat it.

In any case, FNMT systems explicitly model some grammatical information leading to different lexical choices, which might not be captured by the automatic metrics, specially BLEU score. In an attempt to focus on the grammaticality of the FNMT systems, we conducted a qualitative analysis of the outputs.

5.2 High versus low resource conditions

In the previous experiments, we evaluated our factored NMT system **from** English to the highly inflected languages Czech and Latvian. In the following experiments, we experiment with another language pair to translate from Arabic to French. In these experiments, we use Arabic in order to translate from a highly inflected language. The target language is French which is a moderately inflected language. The Arabic→French language pair does not include English which is the most used language to translate and the one with the biggest language resources. Then, translating a language pair without including English is more difficult. Moreover, French and Arabic are not related languages, they do not share morphological roots and either the alphabet. Therefore, it is a challenge to use this language pair. Firstly, we perform experiments under low resource conditions using a small training dataset. Secondly, experiments under high resource conditions adding a large corpus for training are carried out.

Training details

For the training of the models, we used `NMTpy` toolkit. We used tied target embeddings (as in Sections 4.5 and 5.1) sharing the same weights for feedback and output. For the sake of simplicity, normalization of the layers and dropout have not been used. In order to avoid exploding gradients, we clipped the norm of the gradient to be no more than 1. The optimizer used is Adadelta with an initial learning rate of 1. We use Xavier weights initialization.

The WEB test set has been used as development set in order to apply early stopping, validating from the 2nd epoch every 1k updates with a patience of 10. The same vocabulary size has been used for Arabic at input side and French at output side for words and lemmas, which is 30k. All the factors vocabulary is covered by the network.

We also applied the BPE method for this experiment using the formula $30k - \#characters$ to obtain the number of BPE units comparable with the vocabulary of the rest of the systems. The joint vocabulary (see Section 3.1.6 for more details) sharing the BPE tokens for source and target language is not beneficial when the languages use different alphabets. Therefore, we have not applied joint vocabularies BPE model, instead we have used different BPE units for each language. Note that we could have been used a method to unify the alphabets using transliteration in order to avoid the problem. On the other hand, we think that French and Arabic are languages that do not share the same roots and the benefit is harder to glimpse. The same procedure is applied for FBPE model, source words and target lemmas are segmented in subwords. Factors are repeated for each subword to synchronize lemmas and factors sequences.

Factored representations

Table 5.8 presents the different models used with factorised representation. BPE algorithm was also applied for FNMT system (FBPE model) as presented previously, in Section 5.1. Also, interleaved FNMT (IFNMT) model is used in this experiment. Additionally, we introduced a new representation called "IFNMT+2nd output" where the first output predicts lemmas and factors as IFNMT but we add a new second output generating also factors at the same positions of the factors in the first output. When lemmas are generated in the first output, the second output generates *null*. IFNMT+2nd output pretends to better model the factors having the advantages of the two factorized architectures: (1) as FNMT model, the model learns specialized embeddings only for factors in the second output and (2) as IFNMT model, factors are also included jointly with the lemmas in the embeddings of the first output and factors output receives as feedback its corresponding lemma generated in the previous timestep which can help the generation of factors.

The two FNMT models have been tested as well, the one using a single *h2o* layer (see Figure 4.2) and the other using separated *h2o* layers (see Figure 5.1 in page 77).

Model	1st output	2nd output
FNMT	ressembler	v-C-3-p
FBPE	ressembl+ er	v-C-3-p v-C-3-p
IFNMT (Tamchyna et al., 2017)	ressembler v-C-3-p	-
IFNMT+2nd output	ressembler v-C-3-p	null v-C-3-p

Table 5.8: Factored representations at target side of the Arabic to French translation systems.

Test sets

We evaluate the models with three test files in order to compare different use cases. These test sets are provided with multiple references to better evaluate with automatic metrics such as BLEU. Table 5.9 provides information about the different test sets used to evaluate the models.

Test set	#Sent.	#Tokens (AR/FR)	#Unique words (AR/FR)	#Ref.
WEB*	409	10k/∼18k	4.2k/3.7k	4
TEXT	352	10k/∼18k	4.1k/3.6k	2
BROADCAST (2h)	466	14k/∼23k	4.7k/4.1k	4

Table 5.9: Test sets for Arabic to French translation under low resource conditions. Information about number of sentences and references in second and last columns, respectively. Number of tokens and number of unique words for each language are shown in the third and forth columns. *WEB test set has been used for development purposes. French unique words and number of tokens are average numbers of the references.

5.2.1 Low resource conditions

The first experiment consists of translating under low resource conditions using a small training dataset. We mentioned previously in the motivation of Chapter 4 that when the dataset is small, the morphological information can help the translation process (Niehues and Cho, 2017). In addition, the factored system can support larger vocabulary because it can generate words from the lemmas and factors vocabularies, which is a good point when data is sparse. The hyperparameters chosen for this experiment have been adapted to the small size of the dataset. Therefore, we used reduced dimensions for the layers: 512 for the recurrent layers and 300 for the embeddings.

Data and preprocessing

The datasets used for training are News Commentary version 9 and 80 hours of broadcast news. News corresponds to the domain of the corpora. Arabic data has been tokenized with the morphological analyser tool MADA (Habash et al., 2013; Pasha et al., 2014) separating prefixes and suffixes from stems. French data is tokenized with Moses and the morphological analysis is performed by MACAON, as for the experiments in Chapter 4. After filtering sentences longer than 100 tokens, the training dataset only contains 150k sentences, 4.6M words for Arabic and 4.7M words for French. The full vocabulary is still large which is a challenge in machine translation, all the unique words in the training vocabulary are 72k for Arabic MADA tokenized words in stems and affixes in source side, 73k for French words and 43k for French lemmas in target side. The vocabulary size of French factors is 282. The factored model can possible generates 148k words from the 30k size lemma vocabulary and the factors vocabulary.

We evaluate the models with in-domain and out-of-domain test sets in order to compare different scenarios where the difficulty to translate increases. These test sets are provided with multiple references to better evaluate with automatic metrics such as BLEU. Table 5.9 provides information about the different test sets used to evaluate the models. One test set is out-of-domain (WEB) and other two test sets are in-domain (TEXT and BROADCAST).

Results

The results for the Arabic to French translation under low resource conditions are presented in Table 5.10.

	Out-of-domain	**In-domain** (news)	
Model	WEB (dev.)	TEXT	BROADCAST
NMT	13.52	10.15	19.05
BPE	14.49	9.40	18.27
FNMT single *h2o* layer	**16.99**	12.27	**25.93**
FNMT sep. *h2o* layers	14.60	11.06	24.07
IFNMT	15.25	11.81	**26.00**
IFNMT+2nd output	15.89	**12.90**	24.06
FBPE	**17.04**	10.39	23.63

Table 5.10: Results for Arabic to French translation under low resource conditions. Scores are measured in BLEU. The training datasets used are News commentary and 80 hours of broadcast news.

We observe that factored models obtained the highest values for all the test sets. This means that factored models are a good option for low resource conditions. FBPE model reaches the best value for WEB test set followed by FNMT using single

h2o layer with a similar result. We see that IFNMT models have better performance for in-domain test sets. The TEXT test set has better score for IFNMT+2nd output model. The last test set which is BROADCAST obtained better score for the IFNMT model followed by FNMT using single *h2o* layer with a similar result.

BPE model does not obtain very high scores due to the different BPE vocabularies for each language which is not benefiting the model. BPE applied into the factored model (FBPE) does not obtain high scores either for TEXT and BROADCAST. However, FBPE scores for WEB test set are the highest. This can be due to the fact that WEB test set is the development set and the dictionary of BPE units has been created using it. The separated *h2o* layers for FNMT model seems not to help in low resource conditions, the higher number of parameters compared to the single *h2o* layer FNMT is not necessary to learn a small dataset. Interleaved models (IFNMT and IFNMT+2nd output) work better when translating in-domain data. This fact suggests that the FNMT architecture, where lemmas and factors are separated in two outputs, generalizes better and it can improve the performance when translating out-of-domain data sets.

5.2.2 High resource conditions

In this set of experiments, we used the same language pair (Arabic→French) adding more data in order to change the conditions and observe the behaviour of the systems. For that purpose, we added the United Nations corpus to the training data we already used in the previous experiment. Interleaved models (IFNMT and IFNMT+2nd output) are not included anymore because they did not get the best results for out-of-domain data.

Data and training options

The dataset added to the previous presented data (news-commentary and broadcast news) is the United Nations corpus which gives a total of 14M of sentences, 315M of words in Arabic and 350M of words in French. The full vocabulary is large, all the unique words in the training vocabulary are 881k for Arabic MADA tokenized words in stems and affixes in source side, 674k for French words and 511k for French lemmas in target side. The preprocessing of the data was performed similarly to previous experiment.

The training options are the same except that the size of the recurrent layer is increased to 1024 dimensions and the size of the embedding layer to 512 dimensions. The vocabulary size remains 30k. For BPE systems, we use 30k BPE units not using joint vocabularies. BPE method is applied as well for FBPE systems.

Results

Table 5.11 presents the results of adding United Nations corpus.

The results show that FBPE system obtains the best performance for all the test sets: WEB, TEXT and BROADCAST.

Model	**Out-of-domain** WEB (dev.)	**In-domain** (news) TEXT	BROADCAST
NMT	29.33	20.86	35.77
BPE	28.32	20.15	32.47
FNMT single *h2o* layer	28.99	18.36	34.26
FNMT sep. *h2o* layers	27.00	18.74	33.70
FBPE	**29.53**	**21.05**	**36.98**

Table 5.11: Results for Arabic to French translation adding United Nations corpus. Scores are measured in BLEU. The training datasets used are News Commentary V9, 80 hours of broadcast news and United Nations.

BPE system does not perform better again because joint vocabularies option is not used (the vocabularies are separated for source and target due to the different scripts of the languages). On the other hand, FBPE is benefiting from the BPE units to handle the increase of training data.

FNMT systems without BPE units obtain low scores confirming the hypothesis that they perform better when they are applied in low resource conditions. Separated *h2o* layers FNMT system performs better than single *h2o* layer FNMT system when translating TEXT. This confirms that separated *h2o* layers FNMT model, which includes more parameters, is a better option when the training dataset is big. On the other hand, translations of WEB and BROADCAST test sets still obtained better scores using single *h2o* layer FNMT system than separated *h2o* layers FNMT system.

5.3 Qualitative evaluation

In this section, we evaluate qualitatively the outputs of the systems. First of all, we observe the alignments from the attention mechanism comparing BPE and FBPE systems for English-to-Czech translations. Secondly, we present the results of human evaluation in WMT 2017 evaluation campaign. Finally, we translate a special test dataset which takes into account some morphological features.

5.3.1 Attention in factored systems

In a factored NMT setup, the attention mechanism distributes weights across all positions in the input sentence in order to make two predictions, one for lemmas and other for factors, as opposite to BPE system which makes only one prediction. Figure 5.2 is an example showing the alignments from target to source sequence of BPE and FBPE systems when using the ensembles of two English-to-Czech models.

In this sentence, the BPE system (translation on the top) erroneously generates the tense in *"nevyhýbá"* (does not avoid) predicting the present. We can see that

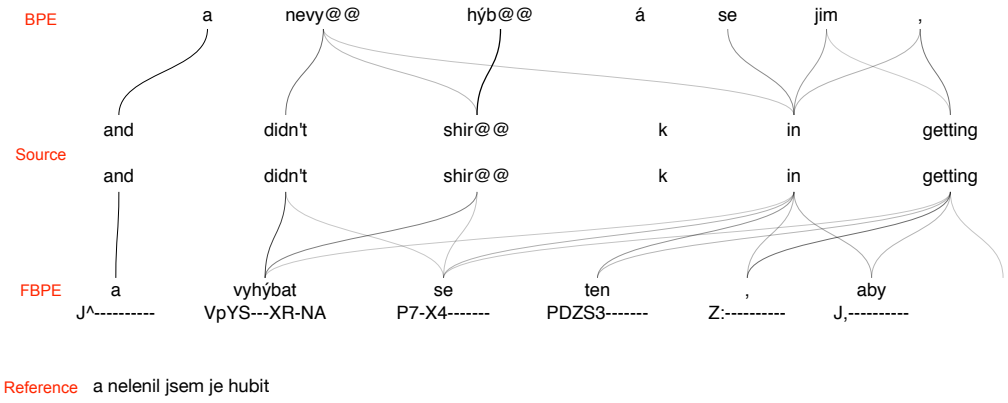

Figure 5.2: An example of attention weight distribution in BPE (top) and FBPE (bottom) output systems aligned to the source sentence (middle) for English-to-Czech translation. Reference sentence corresponds to the last line.

the subword unit *"nevy@@"* in BPE system is rather strongly linked to the source *"didn't"*, which allowed the system to correctly predicts negative polarity. On the other hand, the ending of the verb *"á"* is not linked by attention to this same source word, from which the morphological past tense feature should have been generated. We observe that FBPE system attention aligns the target *"vyhýbat"* and *"VpYS—XR-NA"* to both the source auxiliary *"didn't"* and the first subword unit of the lexical verb *"shirk"* (*"shir@@"*), which enables the successful prediction of the right lemma and factors, *i.e.* negation polarity (N) and past tense (R).

FBPE system models explicitly the grammar at target side. With this information, a source grammatical word (auxiliary, preposition, negative particle, etc.) can be easier aligned correctly, which allows the system to better predict the factors output. We assume that this peculiarity ensures a better grammatical translation adequacy.

5.3.2 Human evaluation

The machine translation systems submitted to the WMT'17 evaluation campaign were evaluated by humans (Bojar et al., 2017). The output translations of the systems were scored with a value between 0 and 100, corresponding to 100 a perfect translation.

The results for human evaluation comparing our submitted BPE and FBPE systems are presented in Table 5.12 for English-to-Czech translation when using ensemble and in Table 5.13 for English-to-Latvian translation when using LM reranking but no ensemble.

There are two types of measures: (1) Average % is the mean of the human annotation scores of the segments for the systems and (2) Average z is as average % but including the standard deviation.

Model	Average %	Average z
BPE	55.2	0.090
FBPE	55.2	0.102

Table 5.12: Official human evaluation results of WMT'17 News translation task for English-to-Czech translation. Systems are ordered by standardized mean DA score.

Model	Average %	Average z
BPE	43.0	-0.198
FBPE	43.2	-0.157

Table 5.13: Official human evaluation results of WMT'17 News translation task for English-to-Latvian translation. Systems are ordered by standardized mean DA score.

Human evaluation shows for English-to-Czech translation task that both systems obtained 55.2% of mean score and FBPE shows a higher score in Average z than BPE system. For English-to-Latvian translation task, BPE system obtained 43% of mean score and FBPE system obtained 43.2% which is a higher score. Also, for Average z, FBPE shows higher score than BPE.

Nevertheless, those differences are not significant, BPE and FBPE systems for both language pairs are in the same cluster of performance according to Wilcoxon rank-sum test at p-level where $p \leq 0.05$.

Appendix A includes the WMT'17 human results for English-to-Czech and English-to-Latvian systems comparing all the submitted systems in the evaluation campaign.

5.3.3 Evaluating morphology

We provide here a scenario to check the benefits of FNMT systems when predicting morphological features. The intuition is that dividing the task of translating a sentence into two easier joint tasks (prediction of lemmas and factors) should help the system to improve the grammar of the predictions. We compare the FBPE system with the BPE system in order to verify the confidence of morphology of each system. Other works previously proposed similar evaluations (Sennrich, 2016).

To this end, we have used the test suite provided by Burlot and Yvon (2017), who propose an evaluation of the morphological competence of a machine translation system performed on an automatically produced test suite. The sentences are all extracted from the English news-2008 corpus provided at WMT removing sentences longer than 15 tokens to better focus on the part of the sentence with the phenomena to analyse. For each source test sentence (the *base*), several *variants* are generated, containing only one difference with the base, and focusing on a

specific morphologically feature on the *target*, *e.g.* person, number, tense, case, etc. Table 5.14 shows an example of the variants of a source test sentence.

Each of her seven children married a Vietnamese, all of them **poor**.
Each of her seven children married a Vietnamese, all of them **rich**.
Each of her seven children married a Vietnamese, all of them **inadequate**.
Each of her seven children married a Vietnamese, all of them **short**.
Each of her seven children married a Vietnamese, all of them **miserable**.

Table 5.14: Example of variants of synonyms and antonyms from the base in the test suite.

A translation is correct when the base and the variants generate a different word and the POS-tag of the morphological feature is correct. This measure is not perfect, some errors could come from the tagger. The good point is that a reference sentence is not needed.

This test suite includes three sets: (1) "A-set": contains contrasts in number for pronouns and nouns, gender for pronouns, tense (future and past) and polarity (if negative) for verbs and comparative for adjectives; (2) "B-set": includes agreements in verbs (number, person and tense), case in nouns and prepositions and lastly, pronouns changed to noun phrases according to gender, number and case; and (3) "C-set": is a selection of nouns, adjectives and verbs replaced by a random hyponym (a subcategory of a word, *e.g.* dog is an animal).

The scoring procedure differs for each set and it is done as follows:

- A-set: the base and variant translations are compared. For example, the base can be the singular number, masculine gender, present tense or affirmative polarity form and the variants are the plural number, feminine gender, plural or past tense or negative polarity forms. If the words which are different contain the correct morphological feature, the translation is considered as good. **Accuracy** of the morphological features is averaged over all the sentences. The A-set corresponds to the evaluation of the grammatical **adequacy** of the translation towards the source.

- B-set: the base and variant translation are compared in order to check two facts: (1) if a pronoun is replaced by a noun phrase and (2) the adjective and the noun in the noun phrase share the same gender number and case. For the coordinated features, some agreements can differ from the base and the variant (*e.g.* gender can be different from the base). For the case of measuring the preposition feature, the case of the first noun on the right of a preposition is checked on the base and variant sentences. Then, the **accuracy** is calculated per feature. The B-set corresponds to the evaluation of the grammatical **fluency** of the generated translations.

- C-set: the **consistency** of morphological features with respect to lexical variations with the same context is evaluated in this set. The average normalised **entropy** is measured on the target sentences. The entropy is null (the best value) when all variants share the same morphological features. By contrast, when entropy is 1, all the sentences in the cluster contain different morphological features. 500 buckets for each feature are considered and each bucket contains 5 sentences.

We first compared the English-to-Czech BPE and FBPE systems **single** models with the seed which reached the best performance without ensembling. The FBPE system includes the nk-reranking (see Table 5.6 for previous results of these models).

Table 5.15 shows the results for the A-set evaluating the adequacy of the translation outputs with respect to the source. The last column corresponds to the mean of all scores. The first two systems correspond to the single models. We can note that BPE system only performs better than FBPE system in the negation of verbs feature. On the other hand, FBPE performs better than BPE detecting the future, the number in nouns and the comparative features. The mean of the scores shows an improvement of +0.5 accuracy in FBPE compared to BPE.

	verbs			**pronouns**		**others**		**mean**
Model	past	future	neg.	fem.	plur.	noun nb.	compar.	
Single								
BPE	89.2%	85.8%	**95.4%**	94.2%	91.4%	73.0%	71.0%	85.7%
FBPE	89.4%	**87.4%**	93.4%	94.8%	91.2%	**75.2%**	**72.0%**	**86.2%**
Ensemble								
BPE	92.0%	**85.0%**	95.4%	92.8%	**92.4%**	78.6%	**74.4%**	87.2%
FBPE	**93.4%**	84.0%	**96.2%**	**94.6%**	91.6%	**80.2%**	73.4%	**87.6%**

Table 5.15: Morphological prediction adequacy (accuracy) English-to-Czech translation (A-set). Single models on the top and ensembling of models on the bottom.

Table 5.16 shows the results for the B-set evaluating the morphological fluency of the translation outputs. The first two systems correspond to the single models. In this case, FBPE outperforms BPE in all the features (+7.6% in the mean). This is due to the explicitly modeling of the target morphology in FBPE system.

The results for the C-set about morphological prediction consistency are shown in Table 5.17. These results are measured in terms of entropy, the scores demonstrate how confident a system is with a morphological feature when lexical variations as synonyms or antonyms are translated. The first two systems correspond to the single models. Again, we see that FBPE system outperforms BPE system in all the features. The results show that FBPE system predicts good morphology even when some variations of the lexicon are introduced.

We also evaluated the ensemble of models for English-to-Czech models. These systems correspond to the last line (the best performance) of each model (BPE and

	coordinated verbs			coord.n	pronouns to nouns			prep.	mean
Model	number	person	tense	case	gender	number	case	case	
Single									
BPE	65.0%	64.0%	59.2%	81.4%	45.8%	47.0%	46.6%	88.8%	62.2%
FBPE	**69.6%**	**67.8%**	**62.0%**	**89.4%**	**59.2%**	**59.6%**	**59.6%**	**91.6%**	**69.8%**
Ensemble									
BPE	**82.2%**	**81.6%**	**75.8%**	**91.0%**	**92.0%**	**92.8%**	**93.0%**	**96.2%**	**88.1%**
FBPE	80.8%	79.6%	75.6%	89.6%	90.6%	90.4%	90.8%	95.8%	86.6%

Table 5.16: Morphological prediction fluency (accuracy) English-to-Czech translation (B-set). Single models on the top and ensembling of models on the bottom.

	nouns	adjectives			verbs				mean
Model	case	gender	number	case	number	person	tense	polarity	
Single									
BPE	.251	.437	.404	.434	.203	.141	.161	.132	.270
FBPE	**.208**	**.401**	**.363**	**.389**	**.152**	**.104**	**.120**	**.090**	**.228**
Ensemble									
BPE	.208	.295	.272	.310	.125	.070	.086	.061	.178
FBPE	.206	**.278**	**.240**	**.269**	.125	.074	.090	.067	**.169**

Table 5.17: Morphological prediction consistency (entropy) English-to-Czech translation (C-set). Single models on the top and ensembling of models on the bottom.

FBPE) in Table 5.6. The results of the morphological evaluation applied on the ensemble of models English-to-Czech translation are shown on the last two systems of the Table 5.15 for the A-set, Table 5.16 for the B-set and Table 5.17 for the C-set.

The results for ensemble of models evaluating the morphology in the translation of Czech show that BPE system performed better than FBPE system only in the B-set. Some features are predicted better with FBPE than BPE in the A-set but the global mean is higher for FBPE system. In the B-set all the features are predicted better with BPE system (+1.5% in the mean). The NMT systems are well-known to produce fluent outputs and when using ensembling the performance improves. The dataset of English-to-Czech translation is very large and it can be enough to well learn the grammatical fluency using BPE and ensembling.

On the other hand, the C-set is better predicted for FBPE than BPE. We observe lower global mean entropy for FBPE than BPE. The features related to verbs and nouns perform similar in the two systems. However, the performance is better for FBPE system in the features related to adjectives. This tends to show that the prediction of the **consistency** is more confident, disregarding lexical variations, when factored approach which explicitly models the morphology is applied.

The morphological evaluation has been applied also for the English-to-Latvian translations on the same manner. For the BPE and FBPE evaluation of the single systems, the models with the seed which reached the best performance without

ensembling and either LM reranking (see Table 5.7). The FBPE system for Latvian also includes the nk-reranking. Table 5.18 shows the results for the A-set, Table 5.19 for the B-set and Table 5.20 for the C-set. The first two systems correspond to single models and the two last systems to ensemble of models.

	verbs		**pronouns**		**nouns**	**mean**
Model	past	future	feminine	plural	number	
Single						
BPE	67.6%	**86.4%**	69.8%	88.6%	69.6%	76.4%
FBPE	**74.0%**	83.4%	**81.4%**	**91.0%**	**70.8%**	**80.1%**
Ensemble						
BPE	71.2%	84.8%	65.0%	86.8%	72.6%	76.1%
FBPE	**73.0%**	**81.2%**	**76.8%**	86.6%	**73.2%**	**78.2%**

Table 5.18: Morphological prediction adequacy (accuracy) English-to-Latvian translation (A-set). Single models on the top and ensembling of models on the bottom.

	coordinated verbs			**coord.n**	**pronouns to nouns**			**prep.**	**mean**
Model	number	person	tense	case	gender	number	case	case	
Single									
BPE	71.0%	60.2%	72.8%	**31.0%**	26.0%	28.2%	26.0%	52.2%	45.9%
FBPE	**76.6%**	**65.2%**	**77.4%**	26.2%	**33.0%**	**33.2%**	**31.4%**	**55.8%**	**49.8%**
Ensemble									
BPE	64.4%	56.6%	65.2%	**40.6%**	38.8%	37.6%	**36.2%**	52.8%	49.0%
FBPE	**78.0%**	**67.0%**	**78.6%**	37.0%	38.6%	**38.0%**	35.6%	**56.0%**	**53.6%**

Table 5.19: Morphological prediction fluency (accuracy) English-to-Latvian translation (B-set). Single models on the top and ensembling of models on the bottom.

We observe similar results when comparing evaluation of translations in Latvian and Czech using a single model. The results for the three sets show a global mean improvement in FBPE system compared to BPE. Only the "future" feature in the A-set and "case" for coordinated nouns in the B-set are better predicted with BPE than FBPE. All the rest of the features are better predicted by FBPE system.

The ensemble of models has been also evaluated for Latvian translations. These systems correspond to the last line (the best performance) of each model (BPE and FBPE) in Table 5.7 applying LM reranking as well.

As difference with Czech translation, in Latvian translation the mean of the three sets are better performing for FBPE than BPE (see Tables 5.18, 5.19 and 5.20). We only see better accuracy for BPE in the case of coordinated nouns feature. We always observe a lower entropy in the C-set with FBPE over all features. Again, another difference with Czech is that English-to-Latvian dataset is 10 times smaller (2M sentences) than English-to-Czech (20M sentences). Therefore, the synchronous

	nouns	adjectives			verbs			mean
Model	case	gender	number	case	number	person	tense	
Single								
BPE	.279	.648	.646	.685	.149	.241	.152	.400
FBPE	**.242**	**.600**	**.599**	**.638**	**.100**	**.173**	**.096**	**.350**
Ensemble								
BPE	.263	.640	.623	.669	.140	.233	.142	.387
FBPE	**.213**	**.608**	**.606**	**.643**	**.099**	**.163**	**.092**	**.346**

Table 5.20: Morphological prediction consistency (entropy) English-to-Latvian translation (C-set). Single models on the top and ensembling of models on the bottom.

learning of lemmas and factors in factored models is helping for the morphology prediction in Latvian translations.

5.4 Conclusion

In this chapter, FNMT systems have been applied in more challenging language pairs: English→Czech, English→Latvian and Arabic→French which include highly inflected languages.

In Section 5.1 about Czech and Latvian translation, we have trained state of the art models applying BPE units in combination of FNMT models, backtranslation, reranking of the hypothesis and ensemble of models. The English-to-Latvian training dataset is large (∼2M) and the English-to-Czech training dataset is much larger (∼20M). We showed the results of the WMT'17 evaluation campaign participation. FNMT models obtained better results in combination with BPE units (FBPE model) in BEER automatic metric without using synthetic data. The Interleaved FNMT (IFNMT) (Tamchyna et al., 2017) model using lemmas and factors in a single output sequence has been trained. We compared it with the rest of the models but it performs only better than word-level NMT model showing that FNMT models using two synchronized outputs have better architecture to learn lemmas and factors. Adding synthetic data, all the systems obtained a big improvement. The reranking method gives more improvement for Latvian translation than Czech translation due to the absence of a linguistic Latvian tool to transform lemmas and factors in words. Having a very strong baseline, FBPE model obtained better results than BPE model in the newstest2017 test set translating Czech when measured in BEER and CTER automatic metrics. BLEU metric is the most common machine translation automatic metric but it does not capture well the morphological advantages of FNMT systems (Ananthakrishnan et al., 2006).

In Section 5.2, Arabic to French translation experiments have been carried out. We have experimented with different FNMT models including FBPE and IFNMT

models. In addition, we introduced a new model which combines IFNMT models with the two synchronized outputs FNMT model. The FNMT systems have been compared with NMT and BPE systems using a small training dataset in a first experiment and a bigger one in a second one. The systems have been evaluated with in-domain and out-of-domain test sets in order to compare different conditions.

We have demonstrated that factored NMT models are a good option for low resource conditions obtaining better results than the rest of the models when using small training dataset and translating in-domain and out-of-domain test sets. IFNMT models obtained better performance with in-domain data than FNMT models but with no big differences. On the other hand, when using out-of-domain data FNMT models perform better than IFNMT. Therefore, FNMT models showed a better architecture than IFNMT models to learn lemmas and factors. FBPE models performed better than FNMT models when training with bigger data showing the benefits of factors and BPE combination.

In Section 5.3, we observed the weights distribution of the alignments extracted from the attention mechanism and showed an example where BPE system could not predict the right tense and FBPE system successfully predict it. Human evaluation in WMT'17 did not result in a clear improvement of FBPE systems due to the lack of annotators in Latvian translation and very similar results in Czech translation.

In a last set of experiments, we measured the morphological features using a special test suite for Czech and Latvian translation. Three test sets have been translated measuring the adequacy, fluency and consistency of the translations in order to prove that FBPE system predicts better morphology than BPE systems. We compared the best FBPE and BPE single systems and an ensemble of them. Mixed results are obtained when translating English-to-Czech, in some cases BPE predict better morphology than FBPE and in other cases the results are the opposite. On the other hand, when translating English-to-Latvian, in all the cases the mean of the features scores obtained by the systems is better for FBPE system than BPE system. These results show that the Latvian morphology is better predicted by FBPE system. One of the reason is that the training dataset is not very big. This demonstrates that FNMT system is suitable when facing low resource conditions.

5.5 Related publications

The related publications of this chapter are:

> Burlot*, F. and García-Martínez*, M., Barrault, L., Bougares, F., and Yvon, F. (2017). Word Representations in Factored Neural Machine Translation. In *Proceedings of the Second Conference on Machine Translation, Volume 1: Research Paper*, pages 20–31, Copenhagen, Denmark. Association for Computational Linguistics. *Equal contribution

> García-Martínez, M., Caglayan, O., Aransa, W., Bardet, A., Bougares, F., and Barrault, L. (2017). LIUM Machine Translation Systems for WMT17 News

Translation Task. In *Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 288–295, Copenhagen, Denmark. Association for Computational Linguistics

CHAPTER 6

CONCLUSION AND PERSPECTIVES

Contents

"Words travel worlds.
Translators do the driving".
Anna Rusconi

6.1 Conclusions

The goal of this research is enhancing neural machine translation (NMT) with additional linguistic information. A new NMT architecture operating at factor level instead of word level has been introduced. In this approach, factors based on linguistic *a priori* knowledge have been used to decompose the words. Various Factored NMT systems have been developed to include several related sequences at input and output sides of the neural network. We have shown that the use of linguistic factors allows the system to improve the generalization ability using lemmas and PoS tags in order to have more coverage with only a small increase of the model complexity. This work is one of the first attempt introducing multiple sequences in order to include additional linguistic information in NMT.

The proposed Factored NMT (FNMT) approach is designed to face some remaining challenges of NMT. Firstly, the limitation on the target vocabulary size which is a consequence of the computationally expensive softmax function at the output layer of the network, leading to a high rate of unknown words. Secondly,

data sparsity which is arising when we face a specific domain or a morphologically rich language. The main feature of the FNMT system is the handling of vocabulary at lemma and linguistic factors (including POS-tags) level, instead of words. The system outputs two symbols synchronously, one for the lemma and the other for the factors. In a post-process, the word is extracted knowing the lemma and the factors. Humans also do this for language generation knowing only the base word and how to inflect it. This feature allows the system to generate new words from a lemma without explicitly seeing all the inflections in the training dataset and without keeping them in the shortlist. We found out that this is specially useful for highly inflected languages and small datasets with low coverage.

In the first experiments, we have compared the FNMT model to the word-level NMT model for English to French translation task. We observed translation improvements using the FNMT system (+1 BLEU point). Moreover, the FNMT system is able to halve the number of generated unknown words. Using a simple `UNK` replacement procedure involving a bilingual dictionary, we are able to obtain even better results (+1.8 BLEU points). The FNMT system is able to model an almost 6 times bigger word vocabulary and increase the coverage of the test file with only a slight increment of the computational cost. The advantage of this approach is that the new generated words are controlled by some linguistic knowledge, that avoids producing incorrect words, as opposed to actual systems using BPE. FNMT approach outperforms some other state of the art systems including BPE applied only in the target language.

Different FNMT architectures have been compared using various dependencies between lemmas and factors and experimenting different kind of feedback to generate the next timestep of the model. We observed the translation outputs in a qualitative analysis perceiving the benefits of the FNMT system over word-level and BPE-level NMT systems. Factors approach is also applied at the input side using English as source language. No benefits have been seen which can be explained by the fact that English is a morphologically poor language and factors do not bring additional information to the word.

More experiments have been performed using morphologically rich languages. English to Czech and English to Latvian WMT'17 data has been used to compare FNMT systems to state of the art systems. We applied BPE units at lemma level and repeating the factors for each subword in a word in the FNMT approach (FBPE system). It performed the best in terms of BEER metric when no backtranslated data is used. Recent state of the art techniques have been applied: backtranslation, n-best reranking and ensembling of models in all the systems pushing the results to be competitive in the WMT'17 evaluation campaign. Having a very strong BPE baseline system, FBPE system outperformed it when translating the WMT'17 official test set to Czech in terms of BEER and CTER automatic metrics.

State of the art models, including BPE, are also compared to the FNMT model using small and big training datasets translating from Arabic to French. We found out that factored models are more robust in low resource conditions. FBPE models performed better than pure FNMT models when trained with big dataset showing

the benefits of combining factors and BPE. Moreover, we experimented with different domains obtaining improvements when the FNMT models are used.

In the last set of experiments, we measured the adequacy, fluency and consistency of the Czech and Latvian translations using a special test suite in order to prove that the FBPE system predicts better morphology than the BPE system. For translation into Czech, factored systems performed better when using single models without ensembling. This is due to the explicit modeling of the target morphology in FBPE system. For translation into Latvian, the results show always a better result for FBPE system, even when using ensembling. One of the reason is that the Latvian training dataset is not as big as the Czech training dataset, therefore, factored system can show its power.

The core finding of this thesis is that linguistic factors in NMT improve the translation performance in low resource conditions. However, the combination of BPE units with FNMT system can benefit models trained in high resource conditions. Lastly, the generation of morphologically correct sentences is improved by adding explicit linguistic information in the target language.

6.2 Perspectives

Factored NMT approach has brought a new field to explore. The additional linguistics in NMT can be applied to any language that incorporates a PoS tagger.

For future work, instead of generating all the factors in the same sequence, the architecture can be extended to produce each factor, independently, in a different sequence. Actually, the factors vocabulary is determined by the training set. If we build the factors vocabulary with each factor separately, the generalization power of the model will be increased. More unseen word forms will be supported. However, the complexity of the model increases and more incorrect factors for a lemma can be generated.

In addition, more types of factors can be included without being necessarily linguistically motivated like the domain.

Further experiments can be performed to check if the factors at input side can benefit morphologically rich languages, when translating from other source languages than English. Including linguistic features at input side is likely to be helpful for generating correct factors. Moreover, the rate of unknown words can be reduced in the source sequence if we use only lemma and factors levels.

Finally, FNMT approach can be explored for other tasks where several related sequences are required. For example, PoS tagging jointly with spoken language understanding tasks. Additionally, multimodal or multilingual machine translation models can be extended with the factored approach adding linguistic information to help the generalization performance.

6.3 Publications

The list of publications related to this thesis are:

García-Martínez, M., Barrault, L., and Bougares, F. (2016). Factored Neural Machine Translation Architectures. In *Proceedings of the International Workshop on Spoken Language Translation*, IWSLT'16, Seattle, USA

García-Martínez, M., Barrault, L., and Bougares, F. (2017). "Neural Machine Translation by Generating Multiple Linguistic Factors". In Camelin, N., Estève, Y., and Martín-Vide, C., editors, *Statistical Language and Speech Processing*, pages 21–31, Cham. Springer International Publishing

García-Martínez, M., Caglayan, O., Aransa, W., Bardet, A., Bougares, F., and Barrault, L. (2017). LIUM Machine Translation Systems for WMT17 News Translation Task. In *Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 288–295, Copenhagen, Denmark. Association for Computational Linguistics

Burlot*, F. and García-Martínez*, M., Barrault, L., Bougares, F., and Yvon, F. (2017). Word Representations in Factored Neural Machine Translation. In *Proceedings of the Second Conference on Machine Translation, Volume 1: Research Paper*, pages 20–31, Copenhagen, Denmark. Association for Computational Linguistics. *Equal contribution

Caglayan, O., García-Martínez, M., Bardet, A., Aransa, W., Bougares, F., and Barrault, L. (2017). NMTPY: A Flexible Toolkit for Advanced Neural Machine Translation Systems. *Prague Bull. Math. Linguistics*, 109:15–28

BIBLIOGRAPHY

Al-Rfou, R., Alain, G., Almahairi, A., Angermüller, C., Bahdanau, D., Ballas, N., Bastien, F., Bayer, J., Belikov, A., Belopolsky, A., Bengio, Y., Bergeron, A., Bergstra, J., Bisson, V., Snyder, J. B., Bouchard, N., Boulanger-Lewandowski, N., Bouthillier, X., de Brébisson, A., Breuleux, O., Carrier, P. L., Cho, K., Chorowski, J., Christiano, P., Cooijmans, T., Côté, M., Côté, M., Courville, A. C., Dauphin, Y. N., Delalleau, O., Demouth, J., Desjardins, G., Dieleman, S., Dinh, L., Ducoffe, M., Dumoulin, V., Kahou, S. E., Erhan, D., Fan, Z., Firat, O., Germain, M., Glorot, X., Goodfellow, I. J., Graham, M., Gülçehre, Ç., Hamel, P., Harlouchet, I., Heng, J., Hidasi, B., Honari, S., Jain, A., Jean, S., Jia, K., Korobov, M., Kulkarni, V., Lamb, A., Lamblin, P., Larsen, E., Laurent, C., Lee, S., Lefrançois, S., Lemieux, S., Léonard, N., Lin, Z., Livezey, J. A., Lorenz, C., Lowin, J., Ma, Q., Manzagol, P., Mastropietro, O., McGibbon, R., Memisevic, R., van Merriënboer, B., Michalski, V., Mirza, M., Orlandi, A., Pal, C. J., Pascanu, R., Pezeshki, M., Raffel, C., Renshaw, D., Rocklin, M., Romero, A., Roth, M., Sadowski, P., Salvatier, J., Savard, F., Schlüter, J., Schulman, J., Schwartz, G., Serban, I. V., Serdyuk, D., Shabanian, S., Simon, É., Spieckermann, S., Subramanyam, S. R., Sygnowski, J., Tanguay, J., van Tulder, G., Turian, J. P., Urban, S., Vincent, P., Visin, F., de Vries, H., Warde-Farley, D., Webb, D. J., Willson, M., Xu, K., Xue, L., Yao, L., Zhang, S., and Zhang, Y. (2016). Theano: A python framework for fast computation of mathematical expressions. *CoRR*, abs/1605.02688.

Alexandrescu, A. (2006). Factored neural language models. In *In HLT-NAACL*.

Allen, R. B. (1987). Several studies on natural language and back-propagation. In *In Proceedings of the IEEE First International Conference on Neural Networks (San Diego, CA), volume n*, pages 335–341. IEEE.

Ananthakrishnan, R., Bhattacharyya, P., Sasikumar, M., and Shah, R. (2006). Some Issues in Automatic Evaluation of English-Hindi MT: More Blues for BLEU. In *Proceeding of 5th International Conference on Natural Language Processing*.

Ataman, D., Negri, M., Turchi, M., and Federico, M. (2017). Linguistically motivated vocabulary reduction for neural machine translation from Turkish to English. In *In Proceedings of EAMT*.

Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*, abs/1409.0473.

Baltescu, P., Blunsom, P., and Hoang, H. (2014). OxLM: A Neural Language Modelling Framework for Machine Translation. *Prague Bull. Math. Linguistics*, 102:81–92.

Belinkov, Y., Durrani, N., Dalvi, F., Sajjad, H., and Glass, J. R. (2017). What do Neural Machine Translation Models Learn about Morphology? *CoRR*, abs/1704.03471.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A Neural Probabilistic Language Model. *J. Mach. Learn. Res.*, 3:1137–1155.

Bilmes, J. A. and Kirchhoff, K. (2003). 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*, NAACL-Short '03, pages 4–6, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bojar, O. (2007). English-to-Czech Factored Machine Translation. In *Proc. of the 2nd WMT*, pages 232–239, Prague, Czech Republic.

Bojar, O., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huang, S., Huck, M., Koehn, P., Liu, Q., Logacheva, V., Monz, C., Negri, M., Post, M., Rubino, R., Specia, L., and Turchi, M. (2017). Findings of the 2017 Conference on Machine Translation (WMT17). In *Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 169–214, Copenhagen, Denmark. Association for Computational Linguistics.

Bojar, O., Graham, Y., Kamran, A., and Stanojević, M. (2016). Results of the WMT16 Metrics Shared Task. In *Proc. WMT*, pages 199–231, Berlin, Germany.

Bojar, O., Jawaid, B., and Kamran, A. (2012). Probes in a Taxonomy of Factored Phrase-based Models. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, WMT '12, pages 253–260, Stroudsburg, PA, USA.

Bojar, O. and Kos, K. (2010). Failures in English-Czech Phrase-based MT. In *Proc. of the 5th WMT*, pages 60–66.

Bojar, O. and Tamchyna, A. (2011). Improving Translation Model by Monolingual Data. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 330–336, Stroudsburg, PA, USA. Association for Computational Linguistics.

Brown, P. F., deSouza, P. V., Mercer, R. L., Pietra, V. J. D., and Lai, J. C. (1992). Class-based N-gram Models of Natural Language. *Comput. Linguist.*, 18(4):467–479.

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). The Mathematics of Statistical Machine Translation: Parameter Estimation. *Comput. Linguist.*, 19(2):263–311.

Burlot, F., Knyazeva, E., Lavergne, T., and Yvon, F. (2016). Two-Step MT: Predicting Target Morphology. In *Proc. IWSLT*, Seattle, USA.

Burlot, F. and Yvon, F. (2017). Evaluating the morphological competence of Machine Translation Systems. In *Proceedings of the Second Conference on Machine Translation (WMT'17)*, Copenhagen, Denmark. Association for Computational Linguistics.

Burlot*, F. and García-Martínez*, M., Barrault, L., Bougares, F., and Yvon, F. (2017). Word Representations in Factored Neural Machine Translation. In *Proceedings of the Second Conference on Machine Translation, Volume 1: Research Paper*, pages 20–31, Copenhagen, Denmark. Association for Computational Linguistics. *Equal contribution.

Caglayan, O., Barrault, L., and Bougares, F. (2016). Multimodal Attention for Neural Machine Translation. *CoRR*, abs/1609.03976.

Caglayan, O., García-Martínez, M., Bardet, A., Aransa, W., Bougares, F., and Barrault, L. (2017). NMTPY: A Flexible Toolkit for Advanced Neural Machine Translation Systems. *Prague Bull. Math. Linguistics*, 109:15–28.

Callison-Burch, C., Osborne, M., and Koehn, P. (2006). Re-evaluating the Role of BLEU in Machine Translation Research. In *In EACL*, pages 249–256.

Castaño, M. A. and Casacuberta., F. (1997). A connectionist approach to machine translation. In *In EUROSPEECH.*

Chen, W., Grangier, D., and Auli, M. (2015). Strategies for Training Large Vocabulary Neural Language Models. *CoRR*, abs/1512.04906.

Chen, W., Matusov, E., Khadivi, S., and Peter, J. (2016). Guided Alignment Training for Topic-Aware Neural Machine Translation. *CoRR*, abs/1607.01628.

Chiang, D. (2005). A Hierarchical Phrase-based Model for Statistical Machine Translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 263–270, Stroudsburg, PA, USA. Association for Computational Linguistics.

Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y. (2014a). On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *Proc. SSST@EMNLP*, pages 103–111, Doha, Qatar.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bougares, F., Schwenk, H., and Bengio, Y. (2014b). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *CoRR*, abs/1406.1078.

Chrisman, L. (1991). Learning Recursive Distributed Representations for Holistic Computation. *CONNECTION SCIENCE*, 3:345–366.

Chung, J., Cho, K., and Bengio, Y. (2016). A Character-level Decoder without Explicit Segmentation for Neural Machine Translation. *CoRR*, abs/1603.06147.

Costa-jussà, R. M. and Fonollosa, R. J. A. (2016). Character-based Neural Machine Translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 357–361.

Crego, J. M. and Yvon, F. (2010). Factored Bilingual N-gram Language Models for Statistical Machine Translation. *Machine Translation*, 24(2):159–175.

Dalvi, F., Durrani, N., Sajjad, H., Belinkov, Y., and Vogel, S. (2017). Understanding and Improving Morphological Learning in the Neural Machine Translation Decoder. In *IJCNLP*.

Devlin, J., Zbib, R., Huang, Z., Lamar, T., Schwartz, R. M., and Makhoul, J. (2014). Fast and Robust Neural Network Joint Models for Statistical Machine Translation. In *ACL*.

Dong, D., Wu, H., He, W., Yu, D., and Wang, H. (2015). Multi-Task Learning for Multiple Language Translation. In *ACL*.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *J. Mach. Learn. Res.*, 12:2121–2159.

Durrani, N., Koehn, P., Schmid, H., and Fraser, A. (2014). Investigating the Usefulness of Generalized Word Representations in SMT. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 421–432, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

El Kholy, A. and Habash, N. (2012a). Rich Morphology Generation Using Statistical Machine Translation. In *Proc. INLG*, pages 90–94.

El Kholy, A. and Habash, N. (2012b). Translate, Predict or Generate: Modeling Rich Morphology in Statistical Machine Translation. In *Proc. EAMT*, pages 27–34, Trento, Italy.

Elliott, D., Frank, S., and Hasler, E. (2015). Multi-Language Image Description with Neural Sequence Models. *CoRR*, abs/1510.04709.

Elman, J. L. (1990). Finding Structure in Time. *Cognitive Science*, 14(2):179–211.

Eriguchi, A., Hashimoto, K., and Tsuruoka, Y. (2016). Tree-to-Sequence Attentional Neural Machine Translation. *CoRR*, abs/1603.06075.

Firat, O. and Cho, K. (2016). Conditional Gated Recurrent Unit with Attention Mechanism. `github.com/nyu-dl/dl4mt-tutorial/blob/master/docs/cgru.pdf`.

Firat, O., Cho, K., Sankaran, B., Yarman Vural, F. T., and Bengio, Y. (2017). Multi-way, Multilingual Neural Machine Translation. *Comput. Speech Lang.*, 45(C):236–252.

Fraser, A., Weller, M., Cahill, A., and Cap, F. (2012). Modeling Inflection and Word-Formation in SMT. In *Proc. EACL*, pages 664–674, Avignon, France.

García-Martínez, M., Barrault, L., and Bougares, F. (2016). Factored Neural Machine Translation Architectures. In *Proceedings of the International Workshop on Spoken Language Translation*, IWSLT'16, Seattle, USA.

García-Martínez, M., Barrault, L., and Bougares, F. (2017). "Neural Machine Translation by Generating Multiple Linguistic Factors". In Camelin, N., Estève, Y., and Martín-Vide, C., editors, *Statistical Language and Speech Processing*, pages 21–31, Cham. Springer International Publishing.

García-Martínez, M., Caglayan, O., Aransa, W., Bardet, A., Bougares, F., and Barrault, L. (2017). LIUM Machine Translation Systems for WMT17 News Translation Task. In *Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 288–295, Copenhagen, Denmark. Association for Computational Linguistics.

Glorot, X. and Bengio, Y. (2010a). 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*.

Glorot, X. and Bengio, Y. (2010b). 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*.

Green, S., Wang, S., Cer, D., and Manning, C. D. (2013). Fast and Adaptive Online Training of Feature-Rich Translation Models. In *ACL*.

Gupta, P., Joshi, N., and Mathur, I. (2014). Automatic Ranking of Machine Translation Outputs Using Linguistic Factors. *International Journal of Advanced Computer Research*, 4(2):510.

Habash, N., Roth, R., Rambow, O., Esk, R., and Tomeh, N. (2013). Morphological Analysis and Disambiguation for Dialectal Arabic. In *In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT.*

Haddow, B. and Koehn, P. (2012). *"Interpolated backoff for factored translation models"*. Association for Machine Translation in the Americas, AMTA.

Hoang, H. and Koehn, P. (2009). Improving Mid-range Reordering Using Templates of Factors. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 372–379, Stroudsburg, PA, USA. Association for Computational Linguistics.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780.

Huck, M., Riess, S., and Fraser, A. (2017a). Target-side word segmentation strategies for neural machine translation. In *In Proceedings of the Second Conference on Machine Translation (WMT)*, Copenhagen, Denmark.

Huck, M., Tamchyna, A., Bojar, O., and Fraser, A. (2017b). Producing Unseen Morphological Variants in Statistical Machine Translation. In *EACL*.

Isozaki, H., Hirao, T., Duh, K., Sudoh, K., and Tsukada, H. (2010). Automatic Evaluation of Translation Quality for Distant Language Pairs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 944–952, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jawaid, B. and Bojar, O. (2014). Two-Step Machine Translation with Lattices. In Chair), N. C. C., Choukri, K., Declerck, T., Loftsson, H., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland. European Language Resources Association (ELRA).

Jean, S., Cho, K., Memisevic, R., and Bengio, Y. (2015a). On Using Very Large Target Vocabulary for Neural Machine Translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10, Beijing, China.

Jean, S., Firat, O., Cho, K., Memisevic, R., and Bengio, Y. (2015b). Montreal Neural Machine Translation Systems for WMT'15. In *WMT@EMNLP*.

Jelinek, F. (1969). Fast Sequential Decoding Algorithm Using a Stack. *IBM J. Res. Dev.*, 13(6):675–685.

Jelinek, F., Bahl, L., and Mercer, R. (1975). Design of a Linguistic Statistical Decoder for the Recognition of Continuous Speech. *IEEE Trans. Inf. Theor.*, 21(3):250–256.

Jordan, M. (1986). *Serial Order: A Parallel Distributed Processing Approach*. ICS report. Institute for Cognitive Science, University of California, San Diego.

Kalchbrenner, N. and Blunsom, P. (2013). Recurrent Continuous Translation Models. In *EMNLP*.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Koehn, P. and Hoang, H. (2007). Factored translation models. In *In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL*, pages 868–876.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical Phrase-based Translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Le, H.-S., Allauzen, A., and Yvon, F. (2012). Continuous Space Translation Models with Neural Networks. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pages 39–48, Stroudsburg, PA, USA. Association for Computational Linguistics.

Le, H.-S., Oparin, I., Messaoudi, A., Allauzen, A., Gauvain, J.-L., and Yvon, F. (2011). Large Vocabulary SOUL Neural Network Language Models. In *INTERSPEECH*.

Lee, J., Cho, K., and Hofmann, T. (2016). Fully Character-Level Neural Machine Translation without Explicit Segmentation. *CoRR*, abs/1610.03017.

Li, J., Cheng, J.-h., Shi, J.-y., and Huang, F. (2012). "Brief Introduction of Back Propagation (BP) Neural Network Algorithm and Its Improvement". In Jin, D. and Lin, S., editors, *Advances in Computer Science and Information Engineering*, pages 553–558, Berlin, Heidelberg. Springer Berlin Heidelberg.

Ling, W., Trancoso, I., Dyer, C., and Black, A. W. (2015). Character-based Neural Machine Translation. *CoRR*, abs/1511.04586.

Luong, M., Le, Q. V., Sutskever, I., Vinyals, O., and Kaiser, L. (2015a). Multi-task Sequence to Sequence Learning. *CoRR*, abs/1511.06114.

Luong, M.-T. and Manning, D. C. (2016). Achieving Open Vocabulary Neural Machine Translation with Hybrid Word-Character Models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1054–1063.

Luong, M.-T., Nakov, P., and Kan, M.-Y. (2010). A Hybrid Morpheme-word Representation for Machine Translation of Morphologically Rich Languages. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 148–157, Stroudsburg, PA, USA. Association for Computational Linguistics.

Luong, T., Sutskever, I., Le, Q. V., Vinyals, O., and Zaremba, W. (2015b). Addressing the Rare Word Problem in Neural Machine Translation. In *ACL*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. In *Proceedings of Workshop at ICLR*.

Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *INTERSPEECH*, volume 2, page 3.

Minkov, E., Toutanova, K., and Suzuki, H. (2007). Generating Complex Morphology for Machine Translation. In *Proc. ACL*, pages 128–135, Prague, Czech Republic.

Mnih, A. and Hinton, G. (2008). A Scalable Hierarchical Distributed Language Model. In *Advances in Neural Information Processing Systems 21*, volume 21, pages 1081–1088.

Nadejde, M., Reddy, S., Sennrich, R., Dwojak, T., Junczys-Dowmunt, M., Koehn, P., and Birch, A. (2017). Syntax-aware Neural Machine Translation Using CCG. *CoRR*, abs/1702.01147.

Nasr, A., Béchet, F., Rey, J.-F., Favre, B., and Roux, J. L. (2011). MACAON, An NLP Tool Suite for Processing Word Lattices. In *Proceedings of the ACL-HLT 2011 System Demonstrations*, pages 86–91.

Neco, R. and Forcada, M. (1997). Asynchronous translations with recurrent neural nets. In *In International Conf. on Neural Networks,*, volume 4, page 2535–2540.

Ngoc-Quan, P., Matthias, S., Elizabeth, S., Thanh-Le, H., Jan, N., and Alexander, W. (2017). KIT's Multilingual Neural Machine Translation systems for IWSLT 2017. In *Proceedings of the International Workshop on Spoken Language Translation*, IWSLT'17, Tokyo, Japan.

Niehues, J. and Cho, E. (2017). Exploiting Linguistic Resources for Neural Machine Translation Using Multi-task Learning. In *Proceedings of the Second Conference on Machine Translation, Volume 1: Research Papers*, pages 80–89, Copenhagen, Denmark. Association for Computational Linguistics.

Niehues, J., Ha, T.-L., Cho, E., and Waibel, A. (2016). Using Factored Word Representation in Neural Network Language Models. In *Proceedings of the First Conference on Machine Translation*, pages 74–82, Berlin, Germany. Association for Computational Linguistics.

Nirenburg, S. (1989). "Knowledge-based machine translation". *Machine Translation*, 4(1):5–24.

Och, F. J. (1999). An Efficient Method for Determining Bilingual Word Classes. In *Proceedings of the Ninth Conference on European Chapter of the Association for Computational Linguistics*, EACL '99, pages 71–76, Stroudsburg, PA, USA. Association for Computational Linguistics.

Och, F. J. and Ney, H. (2004). The Alignment Template Approach to Statistical Machine Translation. *Comput. Linguist.*, 30(4):417–449.

Paikens, P., Rituma, L., and Pretkalnina, L. (2013). Morphological analysis with limited resources: Latvian example. In *Proc. NODALIDA*, pages 267–277.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002a). BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002b). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA.

Pascanu, R., Mikolov, T., and Bengio, Y. (2012). Understanding the exploding gradient problem. *CoRR*, abs/1211.5063.

Pasha, A., Elbadrashiny, M., Diab, M., Elkholy, A., Eskandar, R., Habash, N., Pooleery, M., Rambow, O., and Roth, R. (2014). MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic. pages 1094–1101.

Press, O. and Wolf, L. (2016). Using the Output Embedding to Improve Language Models. *CoRR*, abs/1608.05859.

Ramanathan, A., Choudhary, H., Ghosh, A., and Bhattacharyya, P. (2009). Case Markers and Morphology: Addressing the Crux of the Fluency Problem in English-Hindi SMT. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 800–808, Stroudsburg, PA, USA. Association for Computational Linguistics.

Rousseau, A. (2013). XenC: An Open-Source Tool for Data Selection in Natural Language Processing. *The Prague Bulletin of Mathematical Linguistics*, 100:73–82.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747.

Sagot, B. (2010). The Lefff, a freely available and large-coverage morphological and syntactic lexicon for French. In *7th international conference on Language Resources and Evaluation (LREC 2010)*, Valletta, Malta.

Sánchez-Cartagena, V. M. and Toral, A. (2016). Abu-MaTran at WMT 2016 Translation Task: Deep Learning, Morphological Segmentation and Tuning on Character Sequences. In *WMT*.

Schwenk, H. (2010). Continuous space language models for statistical machine translation. In *The Prague Bulletin of Mathematical Linguistics, (93):137–146.*

Schwenk, H. (2012). Continuous Space Translation Models for Phrase-Based Statistical Machine Translation. In *COLING (Posters)*, pages 1071–1080.

Sennrich, R. (2016). How Grammatical is Character-level Neural Machine Translation? Assessing MT Quality with Contrastive Translation Pairs. *CoRR*, abs/1612.04629.

Sennrich, R. and Haddow, B. (2016). Linguistic Input Features Improve Neural Machine Translation. *CoRR*, abs/1606.02892.

Sennrich, R., Haddow, B., and Birch, A. (2016a). Edinburgh Neural Machine Translation Systems for WMT 16. *CoRR*, abs/1606.02891.

Sennrich, R., Haddow, B., and Birch, A. (2016b). Improving Neural Machine Translation Models with Monolingual Data. In *Proc. ACL*.

Sennrich, R., Haddow, B., and Birch, A. (2016c). Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725. Association for Computational Linguistics.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Stanojević, M. and Sima'an, K. (2014). Fitting Sentence Level Translation Evaluation with Many Dense Features. In *Proc. EMNLP*, pages 202–206, Doha, Qatar.

Straková, J., Straka, M., and Hajič, J. (2014). Open-Source Tools for Morphology, Lemmatization, POS Tagging and Named Entity Recognition. In *Proc. ACL: System Demos*, pages 13–18, Baltimore, MA.

Surafel, M. L., Quintino, F. L., Marco, T., Matteo, N., and Marcello, F. (2017). FBK's Multilingual Neural Machine Translation System for IWSLT 2017. In *Proceedings of the International Workshop on Spoken Language Translation*, IWSLT'17, Tokyo, Japan.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. *CoRR*, abs/1409.3215.

Tamchyna, A., Marco, M. W.-D., and Fraser, A. (2017). Modeling Target-Side Inflection in Neural Machine Translation. In *Proceedings of the Second Conference on Machine Translation (WMT)*, Copenhagen, Denmark.

Toutanova, K., Suzuki, H., and Ruopp, A. (2008). Applying Morphology Generation Models to Machine Translation. In *Proc. ACL-08: HLT*, pages 514–522, Columbus, OH.

Vanden Berghen, F. and Bersini, H. (2005). CONDOR, a New Parallel, Constrained Extension of Powell's UOBYQA Algorithm: Experimental Results and Comparison with the DFO Algorithm. *J. Comput. Appl. Math.*, 181(1):157–175.

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. (2015). Show and tell: A neural image caption generator. In *Computer Vision and Pattern Recognition*.

Wang, W., Peter, J.-T., Rosendahl, H., and Ney, H. (2016). CharacTer: Translation Edit Rate on Character Level. In *Proc. WMT*, pages 505–510, Berlin, Germany.

Weller, M., Fraser, A. M., and im Walde, S. S. (2013). Using subcategorization knowledge to improve case prediction for translation to German. In *ACL (1)*, pages 593–603. The Association for Computer Linguistics.

Weller-Di Marco, M., Fraser, A., and Schulte im Walde, S. (2016). Modeling Complement Types in Phrase-Based SMT. In *Proceedings of the First Conference on Machine Translation*, pages 43–53, Berlin, Germany. Association for Computational Linguistics.

Wu, D. (1997). Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. *Comput. Linguist.*, 23(3):377–403.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., and Dean, J. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR*, abs/1609.08144.

Wu, Y., Yamamoto, H., Lu, X., Matsuda, S., Hori, C., and Kashioka, H. (2012). Factored recurrent neural network language model in TED lecture transcription. In *IWSLT*.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. (2015). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 2048–2057.

Yamada, K. and Knight, K. (2001). A Syntax-based Statistical Translation Model. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, ACL '01, pages 523–530, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yang, Z., Salakhutdinov, R., and Cohen, W. W. (2016). Multi-Task Cross-Lingual Sequence Tagging from Scratch. *CoRR*, abs/1603.06270.

Youssef, I., Sakr, M., and Kouta, M. (2009). Linguistic Factors in Statistical Machine Translation Involving Arabic Language. *CoRR*, 9(11):154–159.

Zeiler, M. D. (2012). ADADELTA: An Adaptive Learning Rate Method. *CoRR*, abs/1212.5701.

Zhang, H., Gildea, D., and Chiang, D. (2008). Extracting Synchronous Grammar Rules from Word-level Alignments in Linear Time. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 1081–1088, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zoph, B. and Knight, K. (2016). Multi-Source Neural Translation. *CoRR*, abs/1601.00710.

APPENDIX A

APPENDIX

A.1 WMT'17 evaluation campaign: human results

The official WMT'17 evaluation campaign was carried out by a human ranking of the translation outputs of the submitted systems. More details can be found in Bojar et al. (2017). In this appendix, we compared our submitted systems (named as LIUM) with the other candidate systems.

#	**Model**	**Average %**	**Average** z
1	uedin-nmt	62.0	0.308
2	*online-B*	*59.7*	*0.240*
3	limsi-factored-norm	55.9	0.111
	LIUM-FBPE	**55.2**	**0.102**
	LIUM-BPE	**55.2**	**0.090**
	CU-Chimera	54.1	0.050
	online-A	*53.3*	*0.029*
8	TT-ufal-8GB	44.9	-0.236
9	TT-afrl-4GB	42.2	-0.315
	PJATK	41.9	-0.327
	TT-base-8GB	40.7	-0.373
	TT-afrl-8GB	40.5	-0.376
13	TT-ufal-4GB	36.5	-0.486
	TT-denisov-4GB	36.6	-0.493

Table A.1: Official human evaluation results of WMT'17 News translation task for English-to-Czech translation. Systems are ordered by standardized mean DA score. Lines between systems indicate clusters according to Wilcoxon rank-sum test at p-level $p \leq 0.05$. Italics font in the model name indicates unconstrained systems (use of resources outside the constraints data provided).

For English-to-Czech translation task (see Table A.1), our LIUM systems are

ranked in the third group. This group contains systems from the 3rd until the 7th ranked system over a total of 14 systems. If we discard unconstrained data systems, this group is between the 2nd and 6th systems over 12 constrained data systems.

#	**Model**	**Average %**	**Average** z
1	*tilde-nc-nmt-smt*	*54.4*	*0.196*
	online-B	*51.6*	*0.121*
	tilde-c-nmt-smt	51.1	0.104
	limsi-fact-norm	50.8	0.075
	usfd-cons-qt21	50.0	0.058
	QT21-Comb	47.1	-0.014
	usfd-cons-kit	47.3	-0.027
	KIT	45.7	-0.063
	uedin-nmt	45.2	-0.072
	tilde-nc-smt	*44.9*	*-0.099*
	LIUM-FBPE	**43.2**	**-0.157**
	LIUM-BPE	**43.0**	**-0.198**
	HY-HNMT	40.1	-0.253
	online-A	*37.5*	*-0.341*
	jhu-pbmt	*36.1*	*-0.368*
	C-3MA	33.3	-0.457
17	PJATK	18.8	-0.947

Table A.2: Official human evaluation results of WMT'17 News translation task for English-to-Latvian translation. Systems are ordered by standardized mean DA score. Lines between systems indicate clusters according to Wilcoxon rank-sum test at p-level $p \leq 0.05$. Italics font in the model name indicates unconstrained systems (use of resources outside the constraints data provided).

For English-to-Latvian translation task (see Table A.2), our LIUM systems are in the first group. However, this group includes all the English-to-Latvian systems in the campaign excepting one, because there were not enough annotators to well evaluate them.

UNIVERSITE BRETAGNE LOIRE

Le Mans Université

Thèse de Doctorat

Mercedes GARCÍA MARTÍNEZ

Traduction Automatique Neuronale Factorisée
Factored Neural Machine Translation

Résumé

La diversité des langues complexifie la tâche de communication entre les humains à travers les différentes cultures. La traduction automatique est un moyen rapide et peu coûteux pour simplifier la communication interculturelle. Récemment, la Traduction Automatique Neuronale (NMT) a atteint des résultats impressionnants. Cette thèse s'intéresse à la Traduction Automatique Neuronale Factorisé (FNMT) qui repose sur l'idée d'utiliser la morphologie et la décomposition grammaticale des mots (lemmes et facteurs linguistiques) dans la langue cible. Cette architecture aborde deux défis bien connus auxquelles les systèmes NMT font face. Premièrement, la limitation de la taille du vocabulaire cible, conséquence de la fonction *softmax*, qui nécessite un calcul coûteux à la couche de sortie du réseau neuronale, conduisant à un taux élevé de mots inconnus. Deuxièmement, le manque de données adéquates lorsque nous sommes confrontés à un domaine spécifique ou une langue morphologiquement riche. Avec l'architecture FNMT, toutes les inflexions des mots sont prises en compte et un vocabulaire plus grand est modélisé tout en gardant un coût de calcul similaire. De plus, de nouveaux mots non rencontrés dans les données d'entraînement peuvent être générés. Dans ce travail, j'ai développé différentes architectures FNMT en utilisant diverses dépendances entre les lemmes et les facteurs. En outre, j'ai amélioré la représentation de la langue source avec des facteurs. Le modèle FNMT est évalué sur différentes langues dont les plus riches morphologiquement. Les modèles à l'état de l'art, dont certains utilisant le *Byte Pair Encoding* (BPE) sont comparés avec le modèle FNMT en utilisant des données d'entraînement de petite et de grande taille. Nous avons constaté que les modèles utilisant les facteurs sont plus robustes aux conditions d'entraînement avec des faibles ressources. Le FNMT a été combiné avec des unités BPE permettant une amélioration par rapport au modèle FNMT entrainer avec des données volumineuses. Nous avons expérimenté avec différents domaines et nous avons montré des améliorations en utilisant les modèles FNMT. De plus, la justesse de la morphologie est mesurée à l'aide d'un ensemble de tests spéciaux montrant l'avantage de modéliser explicitement la morphologie de la cible. Notre travail montre les bienfaits de l'application de facteurs linguistiques dans le NMT.

Mots clés
Traduction Automatique Neuronale, Modèles Factorisés, Apprendissage Profond, Réseaux de Neurones, Traduction Automatique

Abstract

Communication between humans across the lands is difficult due to the diversity of languages. Machine translation is a quick and cheap way to make translation accessible to everyone. Recently, Neural Machine Translation (NMT) has achieved impressive results. This thesis is focus on the Factored Neural Machine Translation (FNMT) approach which is founded on the idea of using the morphological and grammatical decomposition of the words (lemmas and linguistic factors) in the target language. This architecture addresses two well-known challenges occurring in NMT. Firstly, the limitation on the target vocabulary size which is a consequence of the computationally expensive softmax function at the output layer of the network, leading to a high rate of unknown words. Secondly, data sparsity which is arising when we face a specific domain or a morphologically rich language. With FNMT, all the inflections of the words are supported and larger vocabulary is modelled with similar computational cost. Moreover, new words not included in the training dataset can be generated. In this work, I developed different FNMT architectures using various dependencies between lemmas and factors. In addition, I enhanced the source language side also with factors. The FNMT model is evaluated on various languages including morphologically rich ones. State of the art models, some using Byte Pair Encoding (BPE) are compared to the FNMT model using small and big training datasets. We found out that factored models are more robust in low resource conditions. FNMT has been combined with BPE units performing better than pure FNMT model when trained with big data. We experimented with different domains obtaining improvements with the FNMT models. Furthermore, the morphology of the translations is measured using a special test suite showing the importance of explicitly modeling the target morphology. Our work shows the benefits of applying linguistic factors in NMT.

Key Words
Neural Machine Translation, Factored models, Deep Learning, Neural Networks, Machine Translation