SYSTRAN's Pure Neural Machine Translation Systems

Josep Crego, Jungi Kim, Guillaume Klein, Anabel Rebollo, Kathy Yang, Jean Senellart
Egor Akhanov, Patrice Brunelle, Aurélien Coquard, Yongchao Deng, Satoshi Enoue, Chiyo Geiss, Joshua Johanson, Ardas Khalsa, Raoum Khiari, Byeongil Ko, Catherine Kobus, Jean Lorieux, Leidiana Martins, Dang-Chuan Nguyen, Alexandra Priori, Thomas Riccardi, Natalia Segal, Christophe Servan, Cyril Tiquet, Bo Wang, Jin Yang, Dakun Zhang, Jing Zhou

SYSTRAN
firstname.lastname@systrangroup.com

Abstract

Since the first online demonstration of Neural Machine Translation (NMT) by LISA (Bahdanau et al., 2014), NMT development has recently moved from laboratory to production systems as demonstrated by several entities announcing roll-out of NMT engines to replace their existing technologies. NMT systems have a large number of training configurations and the training process of such systems is usually very long, often a few weeks, so role of experimentation is critical and important to share. In this work, we present our approach to production-ready systems simultaneously with release of online demonstrators covering a large variety of languages (12 languages, for 32 language pairs). We explore different practical choices: an efficient and evolutive open-source framework; data preparation; network architecture; additional implemented features; tuning for production; *etc*. We discuss about evaluation methodology, present our first findings and we finally outline further work.

Our ultimate goal is to share our expertise to build competitive production systems for "*generic*" translation. We aim at contributing to set up a collaborative framework to speed-up adoption of the technology, foster further research efforts and enable the delivery and adoption to/by industry of use-case specific engines integrated in real production workflows. Mastering of the technology would allow us to build translation engines suited for particular needs, outperforming current simplest/uniform systems.

1 Introduction

Neural MT has recently achieved state-of-the-art performance in several large-scale translation tasks. As a result, the deep learning approach to MT has received exponential attention, not only by the MT research community but by a growing number of private entities, that begin to include NMT engines in their production systems.

In the last decade, several open-source MT toolkits have emerged—Moses (Koehn et al., 2007) is probably the best-known out-of-the-box MT system—coexisting with commercial alternatives, though lowering the entry barriers and bringing new opportunities on both research and business areas. Following this direction, our NMT system is based on the open-source project `seq2seq-attn`[1] initiated by the Harvard NLP group[2] with the main contributor Yoon Kim. We are contributing to the project by sharing several features described in this technical report, which are available to the MT community.

Neural MT systems have the ability to directly model, in an end-to-end fashion, the association from an input text (in a source language) to its translation counterpart (in a target language). A major strength of Neural MT lies in that all the necessary knowledge, such as syntactic and semantic information, is learned by taking the global sentence context into consideration when modeling translation. However, Neural MT engines are known to be computationally very expensive, sometimes needing for several weeks to accomplish the training phase, even making use of cutting-edge hardware to accelerate computations. Since our interest is for a large variety of languages, and that based on our long experience with machine translation, we do not believe that a one-fits-all approach would work optimally for lan-

[1] `https://github.com/harvardnlp/seq2seq-attn`
[2] `http://nlp.seas.harvard.edu`

guages as different as Korean, Arabic, Spanish or Russian, we did run hundreds of experiments, and particularily explored language specific behaviors. One of our goal would indeed be to be able to inject existing language knowledge in the training process.

In this work we share our recipes and experience to build our first generation of production-ready systems for "generic" translation, setting a starting point to build specialized systems. We also report on extending the baseline NMT engine with several features that in some cases increase performance accuracy and/or efficiency while for some others are boosting the learning curve, and/or model speed. As a machine translation company, in addition to decoding accuracy for "generic domain", we also pay special attention to features such as:

- Training time
- Customization possibility: user terminology, domain adaptation
- Preserving and leveraging internal format tags and misc placeholders
- Practical integration in business applications: for instance online translation box, but also translation batch utilities, post-editing environment...
- Multiple deployment environments: cloud-based, customer-hosted environment or embedded for mobile applications
- *etc*

More important than unique and uniform translation options, or reaching state-of-the-art research systems, our focus is to reveal language specific settings, and practical tricks to deliver this technology to the largest number of users.

The remaining of this report is as follows: Section 2 covers basic details of the NMT system employed in this work. Description of the translation resources are given in section 3. We report on the different experiments for trying to improve the system by guiding the training process in section 4 and section 5, we discuss about performance. In section 6 and 7, we report on evaluation of the models and on practical findings. And we finish by describing work in progress for the next release.

2 System Description

We base our NMT system on the encoder-decoder framework made available by the open-source project `seq2seq-attn`. With its root on a number of established open-source projects such as Andrej Karpathy's char-rnn,[3] Wojciech Zaremba's standard long short-term memory (LSTM)[4] and the rnn library from Element-Research,[5] the framework provides a solid NMT basis consisting of LSTM, as the recurrent module and faithful reimplementations of *global-general-attention* model and *input-feeding* at each time-step of the RNN decoder as described by Luong et al. (2015).

It also comes with a variety of features such as the ability to train with bidirectional encoders and pre-trained word embeddings, the ability to handle unknown words during decoding by substituting them either by copying the source word with the most attention or by looking up the source word on an external dictionary, and the ability to switch between CPU and GPU for both training and decoding. The project is actively maintained by the Harvard NLP group[6].

Over the course of the development of our own NMT system, we have implemented additional features as described in Section 4, and contributed back to the open-source community by making many of them available in the `seq2seq-attn` repository.

`seq2seq-attn` is implemented on top of the popular scientific computing library *Torch*.[7] Torch uses *Lua*, a powerful and light-weight script language, as its front-end and uses the *C* language where efficient implementations are needed. The combination results in a fast and efficient system both at the development and the run time. As an extension, to fully benefit from multi-threading, optimize CPU and GPU interactions, and to have finer control on the objects for runtime (sparse matrix, quantized tensor, ...), we developed a *C*-based decoder using the *C* APIs of *Torch*, called C-torch, explained in detail in section 5.4.

The number of parameters within an NMT model can grow to hundreds of millions, but there are also a handful of meta-parameters that need to be manually determined. For some of the

[3] `https://github.com/karpathy/char-rnn`
[4] `https://github.com/wojzaremba/lstm`
[5] `https://github.com/Element-Research/rnn`
[6] `http://nlp.seas.harvard.edu`
[7] `http://torch.ch`

Model	
	Embedding dimension: 400-1000
	Hidden layer dimension: 300-1000
	Number of layers: 2-4
	Uni-/Bi-directional encoder
Training	
	Optimization method
	Learning rate
	Decay rate
	Epoch to start decay
	Number of Epochs
	Dropout: 0.2-0.3
Text unit	Section 4.1
	Vocabulary selection
	Word vs. Subword (e.g. BPE)
Train data	Section 3
	size (quantity vs. quality)
	max sentence length
	selection and mixture of domains

Table 1: There are a large number of meta-parameters to be considered during training. The optimal set of configurations differ from language pair to language pair.

meta-parameters, many previous work presents clear choices on their effectiveness, such as using the attention mechanism or feeding the previous prediction as input to the current time step in the decoder. However, there are still many more meta-parameters that have different optimal values across datasets, language pairs, and the configurations of the rest of the meta-parameters. In table 1, we list the meta-parameter space that we explored during the training of our NMT systems.

In appendix B, we detail the parameters used for the online system of this first release.

3 Training Resources

Training "generic" engines is a challenge, because there is no such notion of generic translation which is what online translation service users are expecting from these services. Indeed online translation is covering a very large variety of use cases, genres and domains. Also available open-source corpora are domain specific: Europarl (Koehn, 2005), JRC (Steinberger et al., 2006) or MultiUN (Chen and Eisele, 2012) are legal texts, ted talk are scientific presentations, open subtitles (Tiedemann, 2012) are colloquial, *etc.* As a result, the training corpora we used for this release were built by doing a weighted mix all of the available sources. For languages with large resources, we did reduce the ratio of the institutional (Europal, UN-type), and colloquial types – giving the preference to news-type, mix of webpages (like Gigaword).

Our strategy, in order to enable more experiments was to define 3 sizes of corpora for each language pair: a baseline corpus (1 million sentences) for quick experiments (day-scale), a medium corpus (2-5M) for real-scale system (week-scale) and a very large corpora with more than 10M segments.

The amount of data used to train online systems are reported in table 2, while most of the individual experimental results reported in this report are obtained with baseline corpora.

Note that size of the corpus needs to be considered with the number of training periods since the neural network is continuously fed by sequences of sentence batches till the network is considered trained. In Junczys-Dowmunt et al. (2016), authors mention using corpus of 5M sentences and training of 1.2M batches each having 40 sentences – meaning basically that each sentence of the full corpus is presented 10 times to the training. In Wu et al. (2016), authors mention 2M steps of 128 examples for English–French, for a corpus of 36M sentences, meaning about 7 iterations on the complete corpus. In our framework, for this release, we systematically extended the training up to 18 epochs and for some languages up to 22 epochs.

Selection of the optimal system is made after the complete training by calculating scores on independent test sets. As an outcome, we have seen different behaviours for different language pairs with similar training corpus size apparently connected to the language pair complexity. For instance, English–Korean training perplexity still decreases significantly between epoch 13 and 19 while Italian–English perplexity decreases marginally after epoch 10. For most languages, in our set-up, optimal systems are achieved around epoch 15.

We did also some experiment on the corpus size. Intuitively, since NMT systems do not have the memorizing capacity of PBMT engines, the fact that the training use 10 times 10M sentence corpus, or 20 times 5M corpus should not make a huge difference. In one of the experiment, we

Language Pair	Training					Testing						
	#Sents	#Tokens		Vocab		#Sents	#Tokens		Vocab		OOV	
		source	target	source	target		source	target	source	target	source	target
en↔br	2.7M	74.0M	76.6M	150k	213k	2k	51k	53k	6.7k	8.1k	47	64
en↔it	3.6M	98.3M	100M	225k	312k	2k	52k	53k	7.3k	8.8k	66	85
en↔ar	5.0M	126M	155M	295k	357k	2k	51k	62k	7.5k	8.7k	43	47
en↔es	3.5M	98.8M	105M	375k	487k	2k	53k	56k	8.5k	9.8k	110	119
en↔de	2.6M	72.0M	69.1M	150k	279k	2k	53k	51k	7.0k	9.6k	30	77
en↔nl	2.1M	57.3M	57.4M	145k	325k	2k	52k	53k	6.7k	7.9k	50	141
en→ko	3.5M	57.5M	46.4M	98.9k	58.4k	2k	30k	26k	7.1k	11k	0	-
en↔fr	9.3M	220M	250M	558k	633k	2k	48k	55k	8.2k	8.6k	77	63
fr↔br	1.6M	53.1M	47.9M	112k	135k	2k	62k	56k	7.4k	8.1k	55	59
fr↔it	3.1M	108M	96.5M	202k	249k	2k	69k	61k	8.2k	8.8k	47	57
fr↔ar	5.0M	152M	152M	290k	320k	2k	60k	60k	8.5k	8.6k	42	61
fr↔es	2.8M	99.0M	91.7M	170k	212k	2k	69k	64k	8.0k	8.6k	37	55
fr↔de	2.4M	73.4M	62.3M	172k	253k	2k	57k	48k	7.5k	9.0k	59	104
fr→zh	3.0M	98.5M	76.3M	199k	168k	2k	67k	51k	8.0k	5.9k	51	-
ja↔ko	1.4M	14.0M	13.9M	61.9k	55.6k	2k	19k	19k	9.3k	8.5k	0	0
nl→fr	3.0M	74.8M	84.7M	446k	260k	2k	49k	55k	7.9k	7.5k	150	-
fa→en	795k	21.7M	20.2M	166k	147k	2k	54k	51k	7.7k	8.7k	197	-
ja→en	1.3M	28.0M	22.0M	24k	87k	2k	41k	32k	6.2k	7.3k	3	-
zh→en	5.8M	145M	154M	246k	225k	2k	48k	51k	5.5k	6.9k	34	-

Table 2: Corpora statistics for each language pair (iso 639-1 2-letter code, expect for Portuguese Brazilian noted as "br"). All language pairs are bidirectional except nlfr, frzh, jaen, faen, enko, zhen. Columns 2-6 indicate the number of sentences, running words and vocabularies referred to training datasets while columns 7-11 indicate the number of sentences, running words and vocabularies referred to test datasets. Columns 12 and 13 indicate respectively the vocabulary of OOV of the source and target test sets. (*M* stand for milions, *k* for thousands). Since jako and enko are trained using BPE tokenization (see section 4.1), there is no OOV.

compared training on a 5M corpus trained over 20 epochs for English to/from French, and the same 5M corpus for only 10 epochs, followed by 10 additional epochs on additional 5M corpus. The 10M being completely homogeneous. In both directions, we observe that the $5 \times 10 + 5 \times 10$ training is completing with a score improvement of $0.8 - 1.2$ compared to the 5×20 showing that the additional corpus is managing to bring a meaningful improvement. This observation leads to a more general question about how much corpus is needed to actually build a high quality NMT engine (learn the language), the role and timing of diversity in the training and whether the incremental gain could not be substituted by terminology feeding (learn the lexicon).

4 Technology

In this section we account for several experiments that improved different aspects of our translation engines. Experiments range from preprocessing techniques to extend the network with the ability to handle named entities, to use multiple word features and to enforce the attention module to be more like word alignments. We also report on different levels of translation customization.

4.1 Tokenization

All corpora are preprocessed with an in-house toolkit. We use standard token separators (spaces, tabs, etc.) as well as a set of language-dependent linguistic rules. Several kinds of entities are recognized (url and number) replacing its content by the appropriate place-holder. A postprocess is used to detokenize translation hypotheses, where the original raw text format is regenerated following equivalent techniques.

For each language, we have access to language specific tokenization and normalization rules. However, our preliminary experiments showed that there was no obvious gain of using these language specific tokenization patterns, and that some of the hardcoded rules were actually degrading the performance. This would need more investigation, but for the release of our first batch systems, we used a generic tokenization model for most of the languages except Arabic, Chinese and German. In our past experiences with Arabic, separating segmentation of clitics was beneficial, and we retained the same procedure. For German and

Chinese, we used in-house compound splitter and word segmentation models, respectively.

In our current NMT approach, vocabulary size is an important factor that determines the efficiency and the quality of the translation system; a larger vocabulary size correlates directly to greater computational cost during decoding, whereas low coverage of vocabulary leads to severe out-of-vocabulary (OOV) problems, hence lowering translation quality.

In most language pairs, our strategy combines a vocabulary shortlist and a placeholder mechanism, as described in Sections 4.2 and 4.3. This approach, in general, is a practical and linguistically-robust option to addressing the fixed vocabulary issue, since we can take the full advantage of internal manually-crafted dictionaries and customisized user dictionaries (UDs).

A number of previous work such as character-level (Chung et al., 2016), hybrid word-character-based (Luong and Manning, 2016) and subword-level (Sennrich et al., 2016b) address issues that arise with morphologically rich languages such as German, Korean and Chinese. These approaches either build accurate open-vacabulary word representations on the source side or improve translation models' generative capacity on the target side. Among those approaches, subword tokenization yields competitive results achieving excellent vocabulary coverage and good efficiency at the same time.

For two language pairs: enko and jaen, we used source and target sub-word tokenization (BPE, see (Sennrich et al., 2016b)) to reduce the vocabulary size but also to deal with rich morphology and spacing flexibility that can be observed in Korean. Although this approach is very seducing by its simplicity and also used systematically in (Wu et al., 2016) and (Junczys-Dowmunt et al., 2016), it does not have significant side effects (for instance generation of impossible words) and is not optimal to deal with actual word morphology - since the same suffix (*josa* in Korea) depending on the frequency of the word ending it is integrated with, will be splitted in multiple representations. Also, in Korean, these *josa*, are is an "agreement" with the previous syllabus based on their final endings: however such simple information is not explicitely or implicitely reachable by the neural network.

The sub-word encoding algorithm *Byte Pair Encoding* (BPE) described by Sennrich et al. (2016b) was re-implemented in C++ for further speed optimization.

4.2 Word Features

Sennrich and Haddow (2016) showed that using additional input features improves translation quality. Similarly to this work, we introduced in the framework the support for an arbitrary number of discrete word features as additional inputs to the encoder. Because we do not constrain the number of values these features can take at the same time, we represent them with continuous and normalized vectors. For example, the representation of a feature f at time-step t is:

$$x_i^{(t)} = \begin{cases} \frac{1}{n_f} & \text{if } f \text{ takes the } i^{th} \text{ value} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where n_f is the number of possible values the feature f can take and with $x^{(t)} \in \mathbb{R}^{n_f}$.

These representations are then concatenated with the word embedding to form the new input to the encoder.

We extended this work by also supporting additional features on the target side which will be predicted by the decoder. We used the same input representation as on the encoder side but shifted the features sequence compared to the words sequence so that the prediction of the features at time-step t depend on the word at time-step t they annotate. Practically, we are generating feature at time $t + 1$ for the word generated at time t.

To learn these target features, we added a linear layer to the decoder followed by the softmax function and used the mean square error criterion to learn the correct representation.

For this release, we only used case information as additional feature. It allows us to work with a lowercased vocabulary and treat the recasing as a separate problem. We observed that the use of this simple case feature in source and target does improve the translation quality as illustrated in Figure 1. Also, we compared the accuracy of the induced recasing with other recasing frameworks (SRI disamb, and in-house recasing tool based on n-gram language models) and observed that the prediction of case by the NN was higher than using external recaser, which was expected since NN has access to source in addition to the source sentence context, and target sentence history.

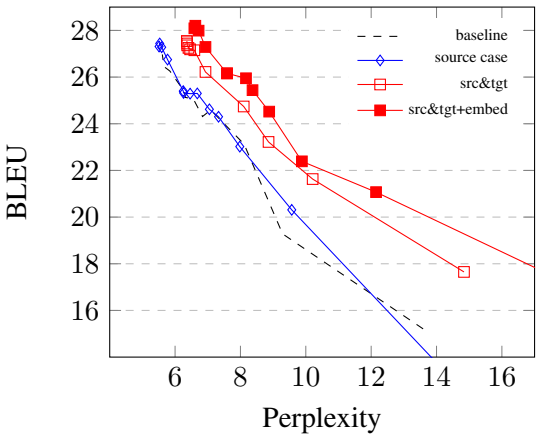

Figure 1: Comparison of training progress (perplexity/BLEU) with/without source (src) and target (tgt) case features, with/without feature embedding (embed) on WMT2013 test corpus for English-French. Score is calculated on lowercase output. The perplexity increases when the target features are introduced because of the additional classification problem. We also notice a noticeable increase in the score when introducing the features, in particular the target features. So these features do not simply help to reduce the vocabulary, but also by themselves help to structure the NN decoding model.

4.3 Named Entities (NE)

SYSTRAN's RBMT and SMT translation engines utilize number and named entity (NE) module to recognize, protect, and translate NE entities. Similarly, we used the same internal NE module to recognize numbers and named entities in the source sentence and temporarily replaced them with their corresponding placeholders (Table 3).

Both the source and the target side of the training dataset need to be processed for NE placeholders. To ensure the correct entity recognition, we cross-validate the recognized entity across parallel dataset, that is: a valid entity recognition in a sentence should have the same type of entity in its parallel pair and the word or phrase covered by the entities need to be aligned to each other. We used `fast_align` (Dyer et al., 2013) to automatically align source words to target words.

In our datasets, generally about one-fifth of the training instances contained one or more NE placeholders. Our training dataset consists of sentences with NE placeholders as well as sentences without them to be able to handle both instantiated and recognized entity types.

NE type	Placeholder
Number	__ent_numeric
Measurement	__ent_numex_measurement
Money	__ent_numex_money
Person (any)	__ent_person
Title	__ent_person_title
First name	__ent_person_firstname
Initial	__ent_person_initials
Last name	__ent_person_lastname
Middle name	__ent_person_middlename
Location	__ent_location
Organization	__ent_organization
Product	__ent_product
Suffix	__ent_suffix
Time expressions	__ent_timex_expression
Date	__ent_date
Date (day)	__ent_date_day
Date (Month)	__ent_date_month
Date (Year)	__ent_date_year
Hour	__ent_hour

Table 3: Named entity placeholder types

Per source sentence, a list of all entities, along with their translations in the target language, if available, are returned by our internal NE recognition module. The entities in the source sentence is then replaced with their corresponding NE placeholders. During beam search, we make sure that an entity placehoder is translated by itself in the target sentence. When the entire target sentence is produced along with the attention weights that provide soft alignments back to the original source tokens, Placeholders in the target sentences are replaced with either the original source string or its translation.

The substitution of the NE placeholders with their correct values needs language pair-specific considerations. In Figure 4, we show that even the handling of Arabic numbers cannot be straightforward as copying the original value in the source text.

4.4 Guided Alignments

We re-reimplemented *Guided alignment* strategy described in Chen et al. (2016). *Guided alignment* enforces the attention weights to be more like alignments in the traditional sense (e.g. IBM4 viterbi alignment) where the word alignments explicitly indicate that source words aligned to a target word are translation equivalents.

Similarly to the previous work, we created an additional criterion on attention weights, L_{ga}, such that the difference in the attention weights and the reference alignments is treated as an error and directly and additionally optimize the output of the

	En → Ko	
	Train	
train data	25 billion	250억
entity-replaced	__ent_numeric billion	__ent_numeric 억
	Decode	
input data	1.4 billion	-
entity-replaced	__ent_numeric billion	-
translated	-	__ent_numeric 억
naive substition	-	1.4억
expected result	-	14억

Table 4: Examples of English and Korean number expressions where naive recognition and substitution fails. Even if the model correctly produces correct placeholders, simply copying the original value will result in incorrect translation. These kind of *structural entities* need language pair-specific treatment.

attention module.

$$L_{ga}(A, \alpha) = \frac{1}{T} \cdot \sum_t \sum_s (A_{st} - \alpha_{st})^2$$

The final loss function for the optimization is then:

$$L_{total} = w_{ga} \cdot L_{ga}(A, \alpha) + (1 - w_{ga}) \cdot L_{dec}(y, x)$$

where A is the alignment matrix, α attention weights, s and t indicating the indices in the source and target sentences, and w_{ga} the linear combination weight for the guided alignment loss.

Chen et al. (2016) also report that decaying w_{ga}, thereby gradually reducing the influence from guided alignment, over the course of training found to be helpful on certain datasets. When guided alignment decay is enabled, w_{ga} is gradually reduced at this rate after each epoch from the beginning of the training.

Without searching for the optimal parameter values, we simply took following configurations from the literature: mean square error (MSE) as loss function, 0.5 as the linear-combination weight for guided alignment loss and the cross-entropy loss for decoder, and 0.9 for decaying factor for guided alignment loss.

For the alignment, we again utilized `fast_align` tool. We stored alignments in sparse format[8] to save memory usage, and for each minibatch a dense matrix is created for faster computation.

Applying such a constraint on attention weights can help locate the original source word more accurately, which we hope to benefits better NE placeholder substitutions and especially unknown word handling.

[8]Compressed Column Storage (CCS)

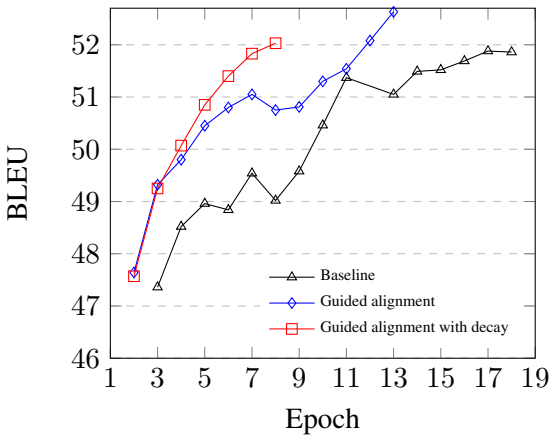

Figure 2: Effect of guided alignment. This particular learning curve is from training an attention-based model with 4 layers of bidirectional LSTMs with 800 dimension on a 5 million French to English dataset.

In Figure 2, we see the effects of guided alignment and its decay rate on our English-to-French generic model. Unfortunately, this full comparative run was disrupted by a power outage and we did not have time to relaunch, however we can still clearly observe that up to the initial 5 epochs, guided alignment, with or without decay, provides rather big boosts over the baseline training. After 5 epochs, the training with decay slow down compare to the training without, which is rather intuitive: the guided alignment is indeed in conflict with the attention learning. What would remain to be seen, is if at the training the training, the baseline and the guided alignment with decay are converging.

4.5 Politeness Mode

Many languages have ways to express politeness and deference towards people being reffered to in sentences. In Indo-European languages, there are two pronouns corresponding to the English You; it is called the *T-V* distinction between the informal Latin pronoun *tu* (**T**) and the polite Latin pronoun *Vos* (**V**). Asian languages, such as Japanese and Korean, make an extensive use of honorifics (respectful words), words that are usually appended to the ends of names or pronouns to indicate the relative ages and social positions of the speakers. Expressing politeness can also impact the vocabu-

lary of verbs, adjectives, and nouns used, as well as sentence structures.

Following the work of Sennrich et al. (2016a), we implemented a politeness feature in our NMT engine: a special token is added to each source sentence during training, where the token indicates the politeness mode observed in the target sentence. Having such an ability to specify the politeness mode is very useful especially when translating from a language where politeness is not expressed, e.g. English, into where such expressions are abundant, e.g. Korean, because it provides a way of customizing politeness mode of the translation output.

Table 5 presents our English-to-Korean NMT model trained with politeness mode, and it is clear that the proper verb endings are generated according to the user selection. After a preliminary evaluation on a small testset from English to Korean, we observed 70 to 80% accuracy of the politeness generation (Table 6). We also noticed that 86% of sentences (43 out of 50) have exactly the same meaning preserved across different politeness modes.

This simple approach, however, comes at a small price, where sometimes the unknown replacement scheme tries to copy the special token in the target generation. A more appropriate approach that we plan to switch to in our future trainings is to directly feed the politeness mode into the sentential representation of the decoder.

4.6 Customization

Domain adaptation is a key feature for our customers—it generally encompasses terminology, domain and style adaptation, but can also be seen as an extension of translation memory for human post-editing workflows.

SYSTRAN engines integrate multiple techniques for domain adaptation, training full new in-domain engines, automatically post-editing an existing translation model using translation memories, extracting and re-using terminology. With Neural Machine Translation, a new notion of "specialization" comes close to the concept of incremental translation as developed for statistical machine translation like (Ortiz-Martínez et al., 2010).

4.6.1 Generic Specialization

Domain adaptation techniques have successfully been used in Statistical Machine Translation. It is well known that a system optimized on a specific text genre obtains higher accuracy results than a "generic" system. The adaptation process can be done before, during or after the training process. Our preliminary experiments follow the latter approach. We incrementally adapt a Neural MT "generic" system to a specific domain by running additional training epochs over newly available in-domain data.

Adaptation proceeds incrementally when new in-domain data becomes available, generated by human translators while post-editing, which is similar to the Computer Aided Translation framework described in (Cettolo et al., 2014).

We experiment on an English-to-French translation task. The generic model is a subsample of the corpora made available for the WMT15 translation task (Bojar et al., 2015). Source and target NMT vocabularies are the 60k most frequent words of source and target training datasets. The in-domain data is extracted from the European Medical Agency (EMEA) corpus. Table 7 shows some statistics of the corpora used in this experiment.

Our preliminary results show that incremental adaptation is effective for even limited amounts of in-domain data (nearly 50k additional words). Constrained to use the original "generic" vocabulary, adaptation of the models can be run in a few seconds, showing clear quality improvements on in-domain test sets.

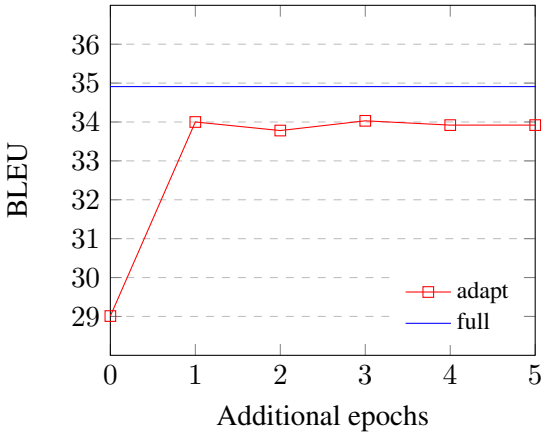

Figure 3: Adaptation with In-Domain data.

Figure 3 compares the accuracy (BLEU) of two systems: full is trained after concatenation of generic and in-domain data; adapt is initially trained over generic data (showing a BLEU score of 29.01 at epoch 0) and adapted after running several training epochs over *only* the in-domain train-

En: A senior U.S. treasury official is urging china to move faster on making its currency more flexible.
Ko with **Formal** mode: 미국 재무부 고위 관계자는 중국이 위안화의 융통성을 더 유연하게 만들기 위해 더 빨리 움직일 것을 촉구했습니다.
Ko with **Informal** mode: 미국 재무부 고위 관계자는 중국이 위안화를 좀더 유연하게 만들기 위해 더 빨리 움직일 것을 촉구하고 있어요.

Table 5: A translation examples from an En-Ko system where the choice of different politeness modes affects the output.

Mode	Correct	Incorrect	Accuracy
Formal	30	14	68.2%
Informal	35	9	79.5%

Table 6: Accuracy of generating correct *Politeness mode* of an English-to-Korean NMT system. The evaluation was carried out on a set of 50 sentences only; 6 sentences were excluded from evaluation because neither the original nor their translations contained any verbs.

Type	*Corpus*	*# lines*	*# src tok (EN)*	*# tgt tok (FR)*
Train	Generic	1M	24M	26M
	EMEA	4,393	48k	56k
Test	EMEA	2,787	23k	27k

Table 7: Data used to train and adapt the generic model to a specific domain. The test corpus also belongs to the specific domain.

ing data. Both systems share the same "generic" NMT vocabularies. As it can be seen the adapt system improves drastically its accuracy after a single additional training epoch, obtaining a similar BLEU score than the full system (separated by .91 BLEU). Note also that each additional epoch using the in-domain training data takes less than 50 seconds to be processed, while training the full system needs more than 17 hours.

Results validate the utility of the adaptation approach. A human post-editor would take advantage of using new training data as soon as it becomes available, without needing to wait for a long full training process. However, the comparison is not entirely fair since full training would allow to include the in-domain vocabulary in the new full model, what surely would result in an additional accuracy improvement.

4.6.2 Post-editing Engine

Recent success of Pure Neural Machine Translation has led to the application of this technology to various related tasks and in particular to the Automatic Post-Editing (APE). The goal of this task

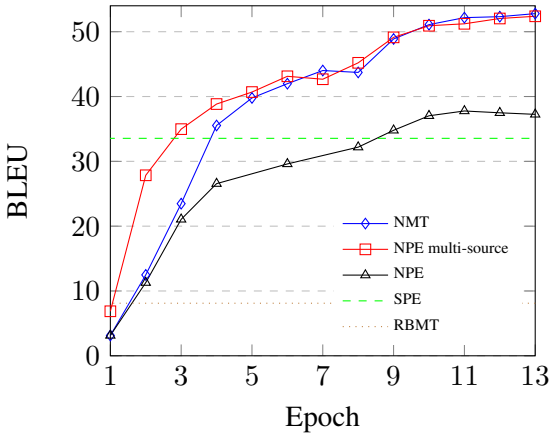

Figure 4: Accuracy results of RBMT, NMT, NPE and NPE multi-source.

is to simulate the behavior of a human post-editor, correcting translation errors made by a MT system.

Until recently, most of the APE approaches have been based on phrase-based SMT systems, either monolingual (MT target to human post-edition) (Simard et al., 2007) or source-aware (Béchara et al., 2011). For many years now SYSTRAN has been offering a hybrid Statistical Post-Editing (SPE) solution to enhance the translation provided by its rule-based MT system (RBMT) (Dugast et al., 2007).

Following the success of Neural Post-Editing (NPE) in the APE task of WMT'16 (Junczys-Dowmunt and Grundkiewicz, 2016), we have run a series of experiments applying the neural approach in order to improve the RBMT system output. As a first experiment, we compared the performance of our English-to-Korean SPE system trained on technical (IT) domain data to two NPE systems trained on the same data: monolingual NPE and multi-source NPE, where the input language and the MT hypothesis sequences have been concatenated together into one input sequence (separated by a special token).

Figure 4 illustrates the accuracy (BLEU) re-

sults of four different systems at different training epochs. The RBMT system performs poorly, confirming the importance of post-editing. Both NPE systems clearly outperform SPE. It can also be observed that adding source information even in a simplest way possible (NPE multi-source), without any source-target alignment, considerably improves NPE translation results.

The system performing NPE multi-source obtains similar accuracy results than pure NMT. What can be seen is that NPE multi-source essentially employs the information from the original sentence to produce translations. However, notice that benefits of utilizing multiple inputs from different sources is clear at earlier epochs while once the model parameters converge, the difference in performances of NMT and NPE multi-source models become negligible.

Further experiments are currently being conducted aiming at finding more sophisticated ways of combining the original source and the MT translation in the context of NPE.

5 Performance

As previously outlined, one of the major drawbacks of NMT engines is the need for cutting-edge hardware technology to face the enormous computational requirements at training and runtime.

Regarding training, there are two major issues: the full training time and the required computation power, i.e. the server investment. For this release, most of our trainings have been running on single GTX GeForce 1080 GPU (about $2.5k) while in (Wu et al., 2016), authors mention using 96 K80 GPU for a full week for training one single language pair (about $250k). On our hardware, full training on 2x5M sentences (see section 3) took a bit less than one month.

A reasonnable target is to maintain training time for any language pair under one week and keeping reasonable investment so that the full research community can have competitive trainings but also indirectly so that all of our customers can benefit from the training technology. To do so, we need to better leverage multiple GPUs on a single server which is on-going engineering work. We also need to continue on exploring how to learn more with less data. For instance, we are convinced that injecting terminology as part of the training data should be competitive with continuing adding full sentences.

Model	BLEU
baseline	49.24
40% pruned	48.56
50% pruned	47.96
60% pruned	45.90
70% pruned	38.38
60% pruned and retrained	**49.26**

Table 8: BLEU scores of pruned models on an internal test set.

Also, shortening training cycle can also be achieved by better control of the training cycle. We have shown that multiple features are boosting the training pace, and if going to bigger network is clearly improving performance. For instance, we are using a bidirectional 4 layer RNN in addition to our regular RNN, but in Wu et al. (2016), authors mention using bidirectional RNN only for the first layer. We need to understand more these phenomena and restrict to the minimum to reduce the model size.

Finally, work on specialization described in sections 4.6.1 and 5.2 are promising for long term maintenance: we could reach a point where we do not need to retrain from scratch but continuously improve existing model and use teacher models to boost initial trainings.

Regarding runtime performance, we have been exploring the following areas and are reaching today throughputs compatible with production requirement not only using GPUs but also using CPU and we report our different strategies in the following sub-sections.

5.1 Pruning

Pruning the parameters of a neural network is a common technique to reduce its memory footprint. This approach has been proven efficient for the NMT tasks in See et al. (2016). Inspired by this work, we introduced similar pruning techniques in `seq2seq-attn`. We reproduced that models parameters can be pruned up to 60% without any performance loss after retraining as shown in Table 8.

With a large pruning factor, neural network's weights can also be represented with sparse matrices. This implementation can lead to lower computation time but more importantly to a smaller memory footprint that allows us to target more environment. Figures 5 and 6 show experiments

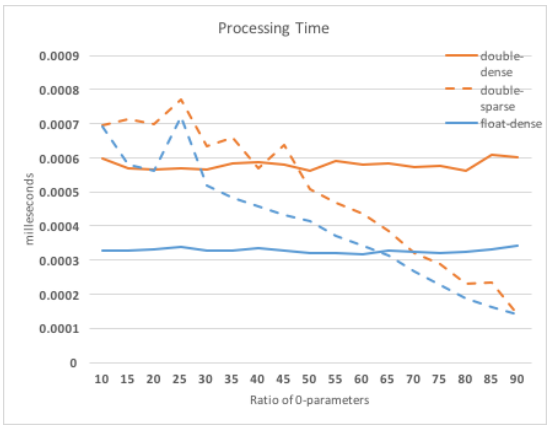

Figure 5: Processing time to perform a 1000×1000 matrix multiplication on a single thread.

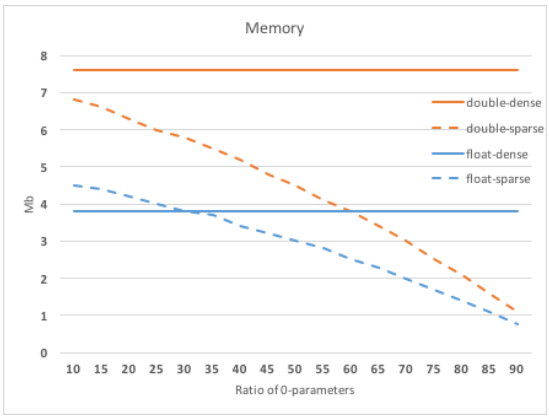

Figure 6: Memory needed to perform a 1000×1000 matrix multiplication.

involving sparse matrices using *Eigen*[9]. For example, when using the `float` precision, a multiplication with a sparse matrix already begins to take less memory when 35% of its parameters are pruned.

Related to this work, we present in Section 5.4 our alternative *Eigen*-based decoder that allows us to support sparse matrices.

5.2 Distillation

Despite that surprisingly accurate, NMT systems need for deep networks in order to perform well. Typically, a 4-layer LSTM with 1000 hidden units per layer (4 x 1000) are used to obtain state-of-the-art results. Such models require cutting-edge hardware for training in reasonable time while inference becomes also challenging on standard setups, or on small devices such as mobile phones. Though, compressing deep models into smaller networks has been an active area of research.

Following the work in (Kim and Rush, 2016), we experimented sequence-level knowledge distillation in the context of an English-to-French NMT task. Knowledge distillation relies on training a smaller student network to perform better by learning the relevant parts of a larger teacher network. Hence, 'wasting' parameters on trying to model the entire space of translations. Sequence-level is the knowledge distillation variant where the student model mimics the teacher's actions at the sequence-level.

The experiment is summarized in 3 steps:

- train a teacher model on a source/reference training set,
- use the teacher model to produce translations for the source training set,
- train a student model on the new source/translation training set.

For our initial experiments, we produced *35*-best translations for each of the sentences of the source training set, and used a normalized n-gram matching score computed at the sentence level, to select the closest translation to each reference sentence. The original training source sentences and their translated hypotheses where used as training data to learn a 2 x 300 LSTM network.

Results showed slightly higher accuracy results for a *70%* reduction of the number of parameters and a *30%* increase on decoding speed. In a second experiment, we learned a student model with the same structure than the teacher model. Surprisingly, the student clearly outperformed the teacher model by nearly 1.5 BLEU.

We hypothesize that the translation performed over the target side of the training set produces a sort of language normalization which is by construction very heterogeneous. Such normalization eases the translation task, being learned by not so deep networks with similar accuracy levels.

5.3 Batch Translation

To increase translation speed of large texts, we support batch translation that works in addition to the beam search. It means that for a beam of size K and a batch of size B, we forward $K \times B$ sequences into the model. Then, the decoder output is split across each batch and the beam path for each sentence is updated sequentially.

[9] `http://eigen.tuxfamily.org`

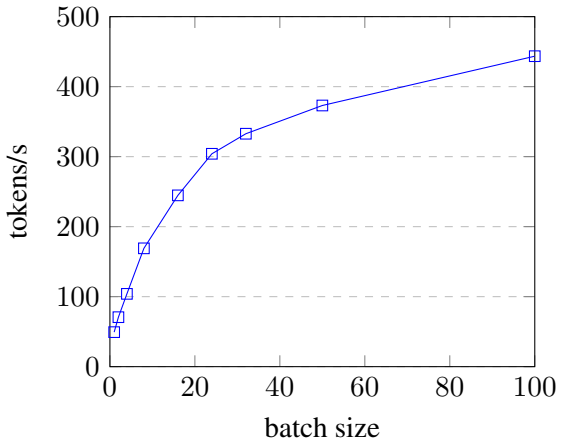

Figure 7: Tokens throughput when decoding from a student model (see section 5.2) with a beam of size 2 and using `float` precision. The experiment was run using a standard *Torch + OpenBLAS* install and 4 threads on a desktop *Intel i7* CPU.

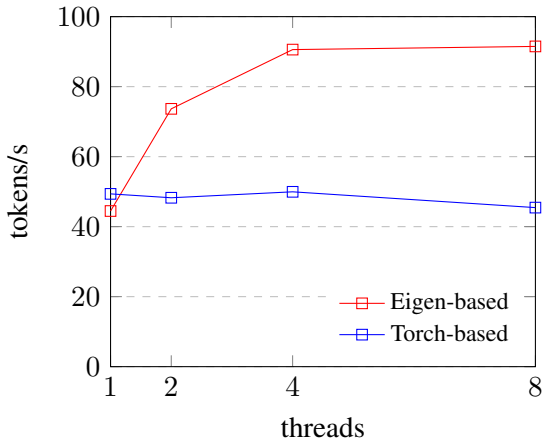

Figure 8: Tokens throughput with a batch of size 1 in the same condition as Figure 7.

As sentences within a batch can have large variations in size, extra care is needed to mask accordingly the output of the encoder and the attention softmax over the source sequences.

Figure 7 shows the speedup obtained using batch decoding in a typical setup.

5.4 C++ Decoder

While *Torch* is a powerful and easy to use framework, we chose to develop an alternative C++ implementation for the decoding on CPU. It increases our control over the decoding process and open the path to further memory and speed improvements while making deployment easier.

Our implementation is graph-based and use *Eigen* for efficient matrix computation. It can load and infer from *Torch* models.

For this release, experiments show that the decoding speed is on par or faster than the *Torch*-based implementation especially in a multi-threaded context. Figure 8 shows the better use of parallelization of the *Eigen*-based implementation.

6 Evaluation

Evaluation of machine translation has always been a challenge and subject to many papers and dedicated workshops (Bojar et al., 2016). While automatic metrics are now used as standard in the research world and have shown good correlation with human evaluation, ad-hoc human evaluation or productivity analysis metrics are rather used in the industry (Blain et al., 2011).

As a translation solution company, even if automatic metrics are used through all the training process (and we give scores in the section 6.1), we care about human evaluation of the results. Wu et al. (2016) mention human evaluation but simultaneously cast a doubt on the referenced human to translate or evaluate. In this context, the claim "*almost indistinguishable with human translation*" is at the same time strong but also very vague. On our side, we have observed during all our experiments and preparation of specialized models, unprecedented level of quality, and contexts where we could claim "super human" translation quality.

However, we need to be very carefully defining the tasks, the human that are being compared to, and the nature of the evaluation. For evaluating technical translation, the nature of the evaluation is somewhat easy and really depending on the user expectation: is the meaning properly conveyed, or is the sentence faster to post-edit than to translate. Also, to avoid doubts about integrity or competency of the evaluation we sub-contracted the task to CrossLang, a company specialized in machine translation evaluation. The test protocol was defined collaboratively and for this first release, we decided to perform ranking of different systems, and we present in the section 6.2 the results obtained on two very different language pairs: English to/from French, and English to Korean.

Finally, in the section 6.3, we also present some qualitative evaluation results showing specificities of the Neural Machine Translation.

6.1 Automatic Scoring and system comparison

Figure 9 plots automatic accuracy results, BLEU, and Perplexities for all language pairs. It is remarkable the high correlation between perplexity and BLEU scores, showing that language pairs with lower perplexity yield higher BLEU scores. Note also that different amounts of training data were used for each system (see Table 2). BLEU scores were calculated over an internal test set.

From the beginning of this report we have used "internal" validation and test sets, what makes it difficult to compare the performance of our systems to other research engines. However, we must keep in mind that our goal is to account for improvements in our production systems. We focus on human evaluations rather than on any automatic evaluation score.

6.2 Human Ranking Evaluation

To evaluate translation outputs and compare with human translation, we have defined the following protocol.

1. For each language pair, 100 sentences "in domain" (*) are collected,
2. These sentences are sent to human translation (**), and translated with candidate model and using available online translation services (***).
3. Without notification of the mix of human and machine translation, a team of 3 professional translators or linguists fluent in both source and target languages is then asked to rank 3 random outputs for each sentence based on their preference as translation. Preference includes accuracy as a priority, but also fluency of the generated translation. They have the choice to give them 3 different ranks, or can also decide to give 2 or the 3 of them the same rank, if they cannot decide.

(*) for Generic domain, sentences from recent news article were selected online, for Technical (IT) sentences part of translation memory defined in section 4.6 were kept apart from the training.
(**) for human translation, we did use translation agency (*human-pro*), and online collaborative translation platform (*human-casual*).
(***) we used Naver Translator[10] (Naver), Google Translate[11] (Google) and Bing Translator[12] (Bing).

For this first release, we experimented on the evaluation protocol for 2 different extremely different categories of language pairs. On one hand, English↔French which is probably the most studied language pairs for MT and for which resources are very large: (Wu et al., 2016) mention about 36M sentence pairs used in their NMT training and the equivalent PBMT is completed by a web-scale target side language models[13]. Also, as English is a low inflected language, the current phrase-based technology for target language English is more competitive due to the relative simplicity of the generation and weight of gigantic language models.

On the other hand, English↔Korean is one of the toughest language pair due to the far distance between English and Korean language, the small availability of training corpus, and the rich agglutinative morphology of Korean. For a real comparison, we ran evaluation against Naver Translation service from English into Korean, where Naver is the main South Korean search engine.

Tables 9 and 10 describe the different evaluations and their results.

Several interesting outcomes:

- Our vanilla English ↦ French model outperforms existing online engines and our best of breed technology.
- For French ↦ English, if the model slightly outperforms (human-casual) and our best of breed technology, it stays behind Google Translate, and more significantly behind Microsoft Translator.
- The generic English ↦ Korean model shows closer results with human translation and outperforms clearly existing online engines.
- The "in-domain" specialized model surprisingly outperforms the reference human translation.

We are aware that far more evaluations are necessary and we will be launching a larger evaluation plan for our next release. Informally, we

[10] http://translate.naver.com

[11] http://translate.google.com

[12] http://translator.bing

[13] In 2007, Google already mentions using 2 trillion words in their language models for machine translation (Brants et al., 2007).

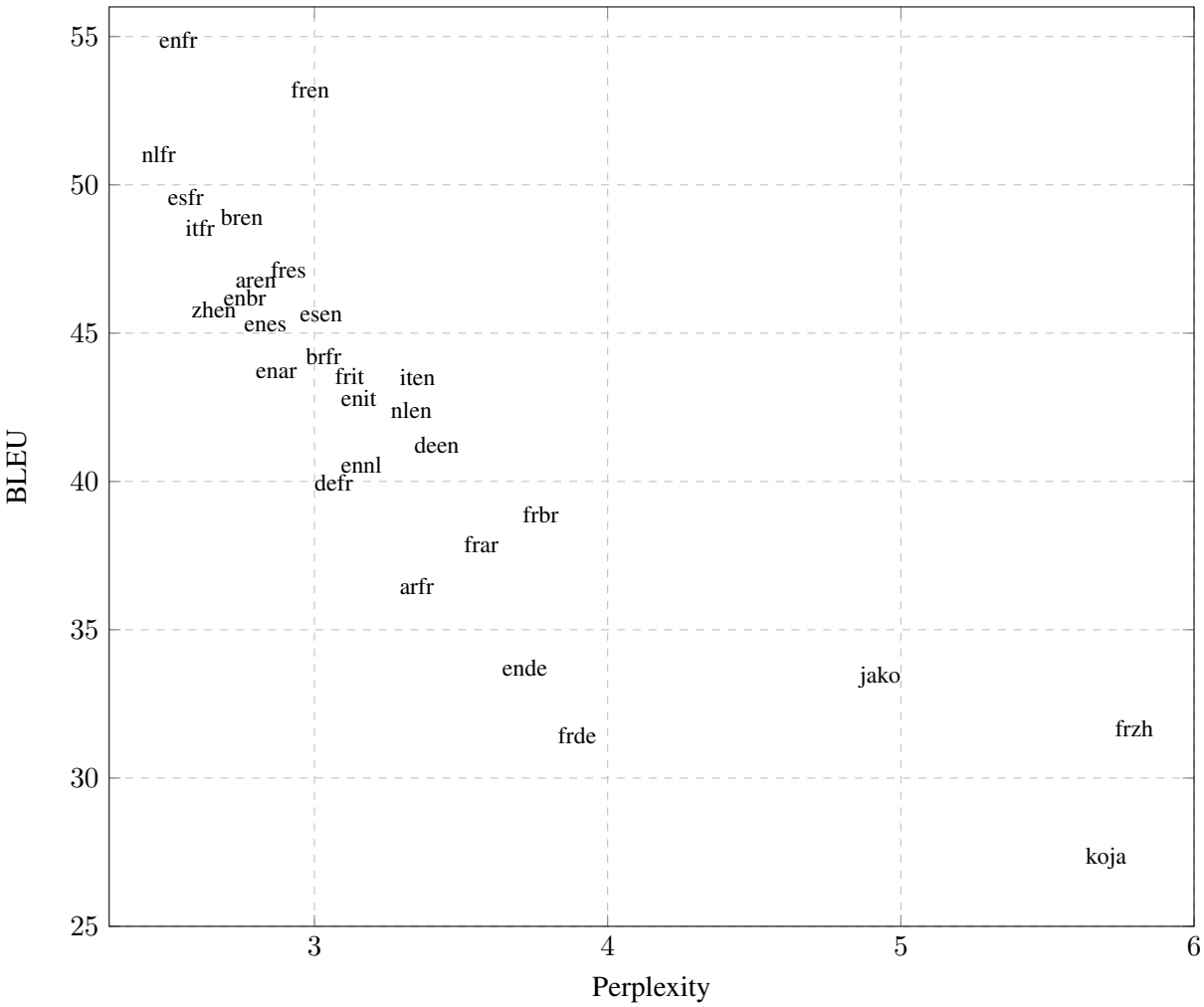

Figure 9: Perplexity and BLEU scores obtained by NMT engines for all language pairs. Perplexity and BLEU scores were calculated respectively on validation and test sets. Note that English-to-Korean and Farsi-to-English systems are not shown in this plot achieving respectively (18.94, 16.50) and (3.84, 34.19) scores for perplexity and BLEU.

Language Pair	**Domain** (*)	**Human Translation** (**)	**Online Translation** (***)
English $\mapsto$ French	Generic	*human-pro*, *human-casual*	Bing, Google
French $\mapsto$ English	Generic	*human-pro*, *human-casual*	Bing, Google
English $\mapsto$ Korean	News	*human-pro*	Naver, Google, Bing
English $\mapsto$ Korean	Technical (IT)	*human-pro*	*N/A*

Table 9: Performed evaluations

do observe that the biggest performance jump are observed on complicated language pairs, like English-Korean or Japanese-English showing that NMT engines are better able to handle major structure differences, but also on the languages with lower resources like Farsi-English demonstrating that NMT is able to learn better with less, and we will explore this even more.

Finally, we are also launching in parallel, a real-life beta-testing program with our customers so that we can also obtain formal feedback from their use-case and related to specialized models.

6.3 Qualitative Evaluation

In the table 11, we report the result of error analysis for NMT, SMT and RBMT for English-French language pair. This evaluation confirms the translation ranking performed in the previous but also exhibits some interesting facts:

- The most salient error comes from missing words or parts of sentence. It is interesting to see though that half of these "omissions" are considered okay by the reviewers and were most of the time not considered as errors - it indeed shows the ability of the system not only to translate but to summarize and get to the point as we would expect from human translation. Of course, we need to fix the cases, where the "omissions" are not okay.
- Another finding is that the engine is badly managing quotes, and we will make sure to specifically teach that in our next release. Other low-hanging fruit are the case generation, which seems sometimes to get off-track, and the handling of Named Entity that we have already introduced in the system but not connected for the release.
- On the positive side, we observe that NMT is drastically improving fluency, reduces slightly meaning selection errors, and handle better morphology although it does not have yet any specific access to morphology (like sub-word embedding). Regarding meaning selection errors, we will focussing on teaching more expressions to the system which is still a major structural weakness compared to PBMT engines.

7 Practical Issues

Translation results from an NMT system, at first glance, is incredibly fluent that your are diverted from its downsides. Over the course of the training and during our internal evaluation, we found out multiple practical issues worth sharing:

1. translating very long sentences
2. translating user input such as a short word or the title of a news article
3. cleaning the corpus
4. alignment

NMT is greatly impacted by the train data, on which NMT learns how to generate accurate and fluent translations holistically. Because the maximal length of a training instance was limited to a certain length during the training of our models, NMT models are puzzled by sentences that exceed this length, not having encountered such a training data. Hard splitting of longer sentences has some side-effects since the model consider both parts as full sentence. As a consequence, whatever is the limit we set for sentence length, we do also need to teach the neural network how to handle longer sentences. For that, we are exploring several options including using separate model based on source/target alignment to find optimal breaking point, and introduce special `<TO BE CONTINUED>` and `<CONTINUING>` tokens. Likewise, very short phrases and incomplete or fragmental sentences were not included in our training data, and consequently NMT systems fail to correctly translate such input texts (e.g. Figure 10). Here also, to enable this, we do simply need to teach the model to handle such input by injecting additional synthetic data.

Also, our training corpus includes a number of resources that are known to contain many noisy data. While NMT seems more robust than other technologies for handling noise, we can still perceive noise effect in translation - in particular for recurring noise. An example is in English-to-Korean, where we see the model trying to systematically convert amount currency in addition to the translation. As demonstrated in Section 5.2, preparing the *right kind of input* to NMT seems to result in more efficient and accurate systems, and such a procedure should also be directly applied to the training data more aggressively.

Finally,let us note that source/target alignment is a must for our users, but this information is missing from NMT output due to soft alignment. To hide this issue from the end-users, multiple alignment heuristic are showing tradictional target-source alignment.

8 Further Work

In this section we outline further experiments currently being conducted. First we extend NMT decoding with the ability to make use of multiple models. Both external models, particularly an n-gram language model, as well as decoding with multiple networks (ensemble). We also work on using external word embeddings, and on modelling unknown words within the network.

8.1 Extending Decoding

8.1.1 Additional LM

As proposed by (Gülçehre et al., 2015), we conduct experiments to integrate an n-gram language model estimated over a large dataset on our Neural MT system. We followed a shallow fusion integration, similar to how language models are used in a standard phrase-based MT decoder.

In the context of beam search decoding in NMT, at each time step t, candidate words x are hypothesized and assigned a score according to the neural network, $p_{NMT}(x)$. Sorted according to their respective scores, the K-best candidates, are reranked using the score assigned by the language model, $p_{LM}(x)$. The resulting probability of each candidate is obtained by the weighted sum of each log-probability $log\ p(x) = log\ p_{LM}(x) + \beta\ log\ p_{NMT}(x)$. Where β is a hyper-parameter that needs to be tuned.

This technique is specially useful for handling out-of-vocabulary words (OOV). Deep networks are technically constrained to work with limited vocabulary sets (in our experiments we use target vocabularies of 60k words), hence suffering from important OOV problems. In contrast, n-gram language models can be learned for very large vocabularies.

Initial results show the suitability of the shallow integration technique to select the appropriate OOV candidate out of a dictionary list (external resource). The probability obtained from a language model is the unique modeling alternative for those word candidates for which the neural network produces no score.

8.1.2 Ensemble Decoding

Ensemble decoding has been verified as a practical technique to further improve the performance compared to a single Encoder-Decoder model (Sennrich et al., 2016b; Wu et al., 2016; Zhou et al., 2016). The improvement comes from the diversity of prediction from different neural network models, which are learned by random initialization seeds and shuffling of examples during training, or different optimization methods towards the development set(Cho et al., 2015). As a consequence, 3-8 isolated models will be trained and ensembled together, considering the cost of memory and training speed. Also, (Junczys-Dowmunt et al., 2016) provides some methods to accelerate the training by choosing different checkpoints as the final models.

We implement ensemble decoding by averaging the output probabilities for each estimation of target word x with the formula:

$$p_x^{ens} = \frac{1}{M} \sum_{m=1}^{M} p_x^m$$

wherein, p_x^m represents probabilities of each x, and M is the number of neural models.

8.2 Extending word embeddings

Although NMT technology has recently accomplished a major breakthrough in Machine Translation field, it still remains constrained due to the limited vocabulary size and to the use of bilingual training data. In order to reduce the negative impact of both phenomena, experiments are currently being held on using external word embedding weights.

Those external word embeddings are not learned by the NMT network from bilingual data only, but by an external model (e.g. word2vec (Mikolov et al., 2013)). They can therefore be estimated from larger monolingual corpora, incorporating data from different domains.

Another advantage lies in the fact that, since external word embedding weights are not modified during NMT training, it is possible to use a different vocabulary for this fixed part of the input during the application or re-training of the model (provided that the weights for the words in new vocabulary come from the same embedding space as the original ones). This may allow a more efficient adaptation to the data coming from a different domain with a different vocabulary.

	Human-pro	Human-casual	Bing	Google	Naver	Systran V8
English ↦ French	-64.2	-18.5	+48.7	+10.4		+17.3
French ↦ English	-56.8	+5.5	-23.1	-8.4		+5
English ↦ Korean	-15.4		+35.5	+33.7	+31.4	+13.2
English ↦ Korean (IT)	+30.3					

Table 10: This table shows relative preference of SYSTRAN NMT compared to other outputs calculated this way: for each triplet where output A and B were compared, we note $pref_{A>B}$ the number of times where A was strictly preferred to B, and E_A the total number of triplet including output A. For each output, E, the number in the table is $compar(\text{SNMT}, E) = (pref_{\text{SNMT}>E} - pref_{E>\text{SNMT}})/E_{\text{SNMT}}$. $compar(\text{SNMT}, E)$ is a percent value in the range $[-1; 1]$

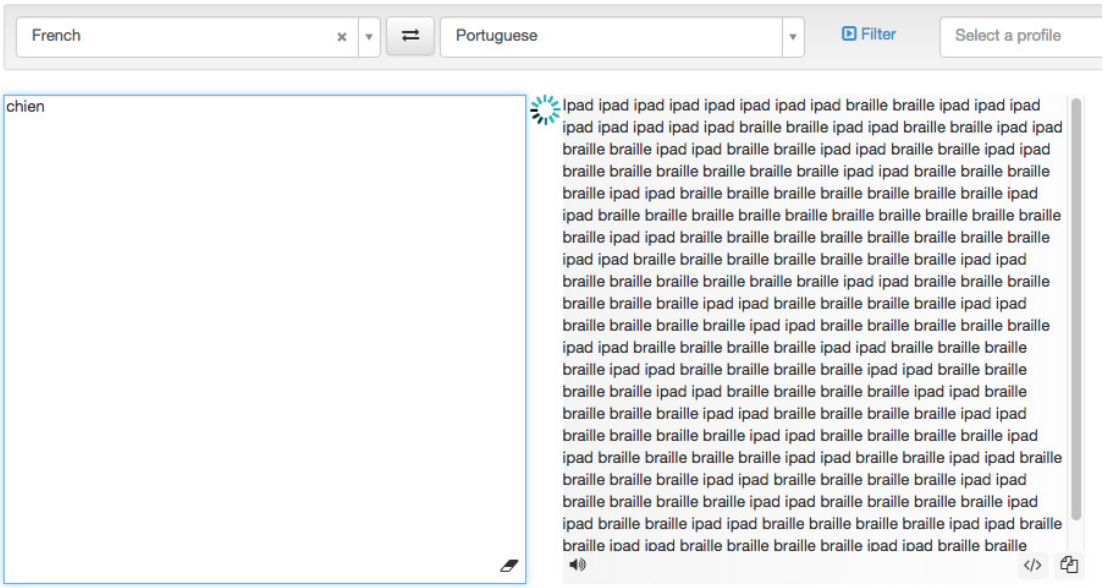

Figure 10: Effect of translation of a single word through a model not trained for that.

Category	NMT	RB	SMT	Example
Entity				
Major	7	5	0	*Galaxy Note 7* ↦ *note 7 de Galaxy* vs. *Galaxy Note 7*
Format	3	1	1	(number localization): *$2.66* ↦ *2.66 $* vs. *2,66 $*
Morphology				
Minor - Local	3	2	3	(tense choice): *accused* ↦ *accusait* vs. *a accusé*
Minor - Sentence Level	3	3	5	*the president [...], she emphasized* ↦ *la président [...], elle a souligné* vs. *la [...], elle*
Major	3	4	6	*he scanned* ↦ *il scanné* vs. *il scannait*
Meaning Selection				
Minor	9	17	7	*game* ↦ *jeu* vs. *match*
Major - Prep Choice	4	9	10	*[... facing war crimes charges] over [its bombardment of ...]* ↦ *contre* vs. *pour*
Major - Expression	3	7	1	*[two] counts of murder* ↦ *chefs de meutre* vs. *chefs d'accusation de meutre*
Major - Not Translated	5	1	4	*he scanned* ↦ *il scanned* vs. *il a scanné*
Major - Contextual Meaning	14	39	14	*33 senior Republicans* ↦ *33 républicains supérieurs* vs. *33 ténors républicains*
Word Ordering and Fluency				
Minor	2	28	15	(determiner):*[without] a [specific destination in mind]* ↦ *sans une destination [...]* vs. *sans destination [...]*
Major	3	16	15	(word ordering):*in the Sept. 26 deaths* ↦ *dans les morts septembre de 26* vs. *dans les morts du 26 septembre*
Missing or Duplicated				
Missing Minor	7	3	1	*a week after the hurricane struck* ↦ *une semaine après l'ouragan* vs. *une semaine après que l'ouragan ait frappé*
Missing Major	6	1	3	*As a working oncologist, Giordano knew [...]* ↦ *Giordano savait* vs. *En tant qu'oncologue en fonction, Giordano savait*
Duplicated Major	2	2	1	*for the Republican presidential nominee* ↦ *au candidat républicain républicain* vs. *au candidat républicain*
Misc. (Minor)				
Quotes, Punctuations	2	0	0	(misplaced quotes) *[...] will be affected by Brexit* ↦ *[...] Sera touchée Par le brexit*
Case	6	0	2	vs. *[...] sera touchée par le Brexit*
Total				
Major	47	84	54	
Minor	36	55	35	
Minor & Major	**83**	**139**	**89**	

Table 11: Human error analysis done for 50 sentences of the corpus defined in the section 6.2 for English-French on NMT, SMT (Google) and RBMT outputs. Error categories are: - issue with entity handling (Entity), issue with Morphology either local or reflecting sentence level missing agreements, issue with Meaning Selection splitted into different sub-categories, - issue with Word Ordering or Fluency (wrong or missing determiner for instance), - missing or duplicated words. Errors are either Minor when reader could still understand the sentence without access to the source, otherwise is considered as Major. Erroneous words are counted in only one category even if several problems add-up - for instance ordering and meaning selection.

8.3 Unknown word handling

When an *unknown word* is generated in the target output sentence, a general encoder-decoder with attentional mechanism utilizes heuristics based on attention weights such that the source word with the most attention is either directly copied as-is or looked up in a dictionary.

In the recent literature (Gu et al., 2016; Gulcehre et al., 2016), researchers have attempted to directly model the unknown word handling within the attention and decoder networks. Having the model learn to take control of both decoding and unknown word handling will result in the most optimized way to address the single unknown word replacement problem, and we are implementing and evaluating the previous approaches within our framework.

9 Conclusion

Neural MT has progressed at a very impressive rate, and it has proven itself to be competitive against online systems trained on train data whose size is several orders of magnitude larger. There is no doubt that Neural MT is definitely a technology that will continue to have a great impact on academia and industry. However, at its current status, it is not without limitations; on language pairs that have abundant amount of monolingual and bilingual train data, phrase-based MT still perform better than Neural MT, because Neural MT is still limited on the vocabulary size and deficient utilization of monolingual data.

Neural MT is not an one-size-fits-all technology such that one general configuration of the model universally works on any language pairs. For example, subword tokenization such as BPE provides an easy way out of the limited vocabulary problem, but we have discovered that it is not always the best choice for all language pairs. Attention mechanism is still not at the satisfactory status and it needs to be more accurate for better controlling the translation output and for better user interactions.

For upcoming releases, we have begun to making even more experiments with injection of various linguistic knowledges, at which SYSTRAN possesses the foremost expertise. We will also apply our engineering know-hows to conquer the practical issues of NMT one by one.

Acknowledgments

We would like to thank Yoon Kim, Prof. Alexander Rush and the rest of members of the Harvard NLP group for their support with the open-source code, their pro-active advices and their valuable insights on the extensions.

We are also thankful to CrossLang and Homin Kwon for their thorough and meaningful definition of the evaluation protocol, and their evaluation team as well as Inah Hong and SunHyung Lee for their work.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473. Demoed at NIPS 2014: http://lisa.iro.umontreal.ca/mt-demo/.

Hanna Béchara, Yanjun Ma, and Josef van Genabith. 2011. Statistical post-editing for a statistical mt system. In *MT Summit*, volume 13, pages 308–315.

Frédéric Blain, Jean Senellart, Holger Schwenk, Mirko Plitt, and Johann Roturier. 2011. Qualitative analysis of post-editing for high quality machine translation. *MT Summit XIII: the Thirteenth Machine Translation Summit [organized by the] Asia-Pacific Association for Machine Translation (AAMT)*, pages 164–171.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Carolina Scarton, Lucia Specia, and Marco Turchi. 2015. Findings of the 2015 workshop on statistical machine translation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 1–46, Lisbon, Portugal, September.

Ondřej Bojar, Yvette Graham, Amir Kamran, and Miloš Stanojević. 2016. Results of the wmt16 metrics shared task. In *Proceedings of the First Conference on Machine Translation*, Berlin, Germany, August.

Thorsten Brants, Ashok C Popat, Peng Xu, Franz J Och, and Jeffrey Dean. 2007. Large language models in machine translation. In *In Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Citeseer.

Mauro Cettolo, Nicola Bertoldi, Marcello Federico, Holger Schwenk, Loic Barrault, and Chrstophe Servan. 2014. Translation project adaptation for mt-enhanced computer assisted translation. *Machine Translation*, 28(2):127–150, October.

Yu Chen and Andreas Eisele. 2012. Multiun v2: Un documents with multilingual alignments. In *LREC*, pages 2500–2504.

Wenhu Chen, Evgeny Matusov, Shahram Khadivi, and Jan-Thorsten Peter. 2016. Guided alignment training for topic-aware neural machine translation. *CoRR*, abs/1607.01628v1.

Sébastien Jean Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007*.

Junyoung Chung, Kyunghyun Cho, and Yoshua Bengio. 2016. A character-level decoder without explicit segmentation for neural machine translation. *CoRR*, abs/1603.06147.

Loïc Dugast, Jean Senellart, and Philipp Koehn. 2007. Statistical Post-Editing on SYSTRAN's Rule-Based Translation System. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 220–223, Prague, Czech Republic, June. Association for Computational Linguistics.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia, June.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany, August. Association for Computational Linguistics.

Çaglar Gülçehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loïc Barrault, Huei-Chi Lin, Fethi Bougares, and Holger Schwenk andYoshua Bengio. 2015. On using monolingual corpora in neural machine translation. *CoRR*, abs/1503.03535.

Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 140–149, Berlin, Germany, August. Association for Computational Linguistics.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2016. Log-linear combinations of monolingual and bilingual neural machine translation models for automatic post-editing. *CoRR*, abs/1605.04800.

M. Junczys-Dowmunt, T. Dwojak, and H. Hoang. 2016. Is Neural Machine Translation Ready for Deployment? A Case Study on 30 Translation Directions. *CoRR*, abs/1610.01108, October.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. *CoRR*, abs/1606.07947.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, pages 79–86.

Minh-Thang Luong and Christopher D. Manning. 2016. Achieving open vocabulary neural machine translation with hybrid word-character models. *CoRR*, abs/1604.00788.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Daniel Ortiz-Martínez, Ismael García-Varea, and Francisco Casacuberta. 2010. Online learning for interactive statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 546–554. Association for Computational Linguistics.

Abigail See, Minh-Thang Luong, and Christopher D Manning. 2016. Compression of neural machine translation models via pruning. In *the proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL2016)*, Berlin, Germany, August.

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. *CoRR*, abs/1606.02892.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Controlling politeness in neural machine translation via side constraints. In *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 35–40, San Diego, California, USA, June. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. 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, Berlin, Germany, August. Association for Computational Linguistics.

Michel Simard, Cyril Goutte, and Pierre Isabelle. 2007. Statistical phrase-based post-editing. In *In Proceedings of NAACL*.

Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaz Erjavec, Dan Tufis, and Dániel Varga. 2006. The jrc-acquis: A multilingual aligned parallel corpus with 20+ languages. *arXiv preprint cs/0609058*.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *LREC*, pages 2214–2218.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. Technical report, Google.

Jie Zhou, Ying Cao, Xuguang Wang, Peng Li, and Wei Xu. 2016. Deep recurrent models with fast-forward connections for neural machine translation. *Transactions of the Association for Computational Linguistics*, 4:371–383.

A Remarkable Results

In this section, we highlight a serie of "remarkable" translations (positively remarkable and also few negatively outstanding) that we found out during evaluation for a variety of languages.

Language Pair	Source	NMT	Google Translate (2016/10/15)
en↦fr	The report also said that although three-quarters of Chinese say their country is playing a larger role in the world than it did a decade ago, most want their government to focus on domestic issues rather than helping other nations.	Le rapport dit aussi que, bien que les trois quarts des Chinois disent que leur pays joue un rôle plus important dans le monde qu'il ne l'a fait il y a dix ans, la plupart veulent que leur gouvernement se concentre sur les questions nationales plutôt que d'aider d'autres nations.	Le rapport indique également que, bien que les trois quarts des Chinois disent que leur pays joue un rôle plus important dans le monde qu'il y a dix ans, la plupart veulent que leur gouvernement de se concentrer sur les questions intérieures plutôt que d'aider les autres nations.
fr↦en	Fenêtre à rue se fermait mal, très très bruyant et mal dormi. Petit déjeuner banal.	The room was very clean and the room was very clean.	street window will not close, very very noisy and slept badly. mundane breakfast.
en↦ko	Forty Republican senators and congressmen have revoked their support for the Republican presidential nominee — with nearly 30 of them calling on him to quit the race altogether in recent days.	40 명의 공화당 상원의원들과 하원의원들이 공화당 대통령 후보자에 대한 그들의 지지를 철회했고, 거의 30 명의 사람들이 최근 며칠 동안 그 경주를 그만두도록 그에게 요청했습니다.	(From Naver NMT[14]): 40여명의 공화당 의원들과 하원 의원들은 거의 30명의 공화당 대통령 후보들을 철회했고, 그들 중 거의 30명이 최근에 그를 은퇴 직전에 그만두게 할 것을 촉구했다.
en↦it	all this leads to feelings of insecurity and conflict thus reinforcing negative spirals.	tutto ciò porta a sentimenti di insicurezza e di conflitto, rafforzando così la spirale negativa.	tutto ciò genera elementi di insicurezza e di conflitto rinforzando quindi spirali negative.
it↦fr	Mi sembra che ciò non abbia nulla a che vedere con una penuria alimentare: si tratta di una vera carestia, di una vera e propria catastrofe che non è naturale ma strutturale, provocata da un regime folle, criminale, psicopatico, peggiore dei peggiori regimi esistiti nell'Unione Sovietica e nella Romania di Ceaucescu.	Il me semble que cela n'a rien à voir avec une pénurie alimentaire: il s'agit d'une véritable famine, d'une véritable catastrophe qui n'est pas naturelle mais structurelle, provoquée par un régime insensé, criminel, psychopathe, pire que les pires régimes de l'union soviétique et de la Roumanie de Ceaucescu.	Il me semble que cela n'a rien à voir avec une pénurie alimentaire: il est une véritable famine, une catastrophe qui est pas naturel, mais une structure, provoquée par un régime criminel fou, psychopathe, le pire des pires régimes existait l'Union soviétique et en Roumanie de Ceaucescu.
it↦fr	Ebbene io sono in totale disaccordo: lo Stato non deve passare in secondo piano; deve invece intervenire di più e garantire una maggiore vigilanza in ambito economico.	je suis tout à fait d'accord: l'état ne doit pas passer sous silence; il doit plutôt agir et assurer une plus grande vigilance dans le domaine économique.	Eh bien, je suis en désaccord total: l'Etat ne doit pas prendre un siège arrière; Il doit plutôt agir plus et assurer une plus grande supervision dans l'économie.
es↦en	¿Es posible un Bangladesh más amable con los niños?	Is Bangladesh more friendly with children?	Can a kinder Bangladesh with children?
en↦de	On the other hand, businesses are constantly complaining about the shortage of specialist staff, using this supposed shortage as an excuse to use cheap labour.	Andererseits beschweren sich Unternehmen ständig über den Mangel an Fachpersonal, indem sie diese angebliche Knappheit als Vorwand benutzen, billige Arbeitskräfte einzusetzen.	Auf der anderen Seite beschweren sich Unternehmen ständig über den Mangel an Fachkräften, wobei diese angebliche Mangelhaftigkeit als Ausrede für billige Arbeitskräfte verwendet wird.
zh↦en	朝天下,在又到了我的互.	Morning News, now it's our interactive time.	(From Google GNMT): North Korea heard the world, and now to our interaction time friends.

Language Pair	Source	NMT	Google Translate (2016/10/15)
br↦en	Face às decisões de Nice, tínhamos de ter em conta esta situação e de adotar uma metodologia que contemplasse as transformações necessárias.	Faced with the Nice decisions, we had to take this situation into account and adopt a methodology that would address the necessary changes.	Nice view of the decisions we had to take account of this situation and adopt a methodology that took into consideration the necessary changes.
fr↦br	Une information présentée au moment opportun signifie la transparence, laquelle crée la confiance et évite à l'entreprise de subir des pertes.	A informação apresentada no momento oportuno significa transparência, que cria confiança e evita que a empresa sofra perdas.	Informação apresentada em uma transparência meio oportuna, que cria confiança e evita a empresa a sofrer perdas.

B Online System Parameters

All systems were trained with 4 LSTM layers, size of word embedding vectors was 500, dropout was set to 0.3 and we used bidirectional RNN (BRNN). Column Guided Alignment indicates wether the network was trained with guided alignments and on which epoch the feature was stopped.

	Tokenization	RNN size	Optimal Epoch	Guided Alignment	NER aware	Special
zh↦en	word boundary-generic	800	12	epoch 4	yes	
en↦it	generic	800	16	epoch 4	yes	
it↦en	generic	800	16	epoch 4	yes	
en↦ar	generic-crf	800	15	epoch 4	no	
ar↦en	crf-generic	800	15	epoch 4	no	
en↦es	generic	800	18	epoch 4	yes	
es↦en	generic	800	18	epoch 4	yes	
en↦de	generic-compound splitting	800	17	epoch 4	yes	
de↦en	compound splitting-generic	800	18	epoch 4	yes	
en↦nl	generic	800	17	epoch 4	no	
nl↦en	generic	800	14	epoch 4	no	
en↦fr	generic	800	18	epoch 4	yes	double corpora ($2\ x\ 5$M)
fr↦en	generic	800	17	epoch 4	yes	
ja↦en	bpe	800	11	no	no	
fr↦br	generic	800	18	epoch 4	yes	
br↦fr	generic	800	18	epoch 4	yes	
en↦pt	generic	800	18	epoch 4	yes	
br↦en	generic	800	18	epoch 4	yes	
fr↦it	generic	800	18	epoch 4	yes	
it↦fr	generic	800	18	epoch 4	yes	
fr↦ar	generic-crf	800	10	epoch 4	no	
ar↦fr	crf-generic	800	15	epoch 4	no	
fr↦es	generic	800	18	epoch 4	yes	
es↦fr	generic	800	18	epoch 4	yes	
fr↦de	generic-compound splitting	800	17	epoch 4	no	
de↦fr	compound splitting-generic	800	16	epoch 4	no	
nl↦fr	generic	800	16	epoch 4	no	
fr↦zh	generic-word segmentation	800	18	epoch 4	no	
ja↦ko	bpe	1000	18	no	no	
ko↦ja	bpe	1000	18	no	no	
en↦ko	bpe	1000	17	no	no	politeness
fa↦en	basic	800	18	yes	no	

Table 12: Parameters for online systems.