Interactive Neural Translation Assistance for Human Translators

Thijs Scheepers
Label305 B.V.
thijs@label305.com

Philip Schulz
ILLC, University of Amsterdam
p.schulz@uva.nl

Abstract

We present the first user study on neural interactive translation prediction. Our neural translation aid was built into existing online translation software. Depending on the prefix already typed by the user the system suggests a list of continuations. We assessed the impact of the system on human translators in a user study with both professional and non-professional translators. The analysis was done with respect to translation speed and translation accuracy as assessed by human judges. We find that our neural translation aid enables relatively fast translations and does not compromise on quality.

1 Introduction

Modern automatic machine translation systems are not yet able to consistently produce high quality document translations. The introduction of neural machine translation (NMT) by Cho et al. (2014) has been a catalyst for some large performance jumps. However, today we still turn to professional human translators for our critical translation tasks.

Taking inspiration from neural suggestion systems used for search queries (Sordoni et al., 2015) and smartphone keyboards (Alsharif et al., 2015) we ask the question: Can we use NMT to create such a suggestion system for human translators?

In this paper, we present an interactive translation prediction system designed to aid human translators. While the translator types, the system provides a list of suggestions for the next word. It takes into account the original sentence and the incomplete translation prefix already entered. The model that powers this system is a two-layer encoder-decoder NMT model with attention

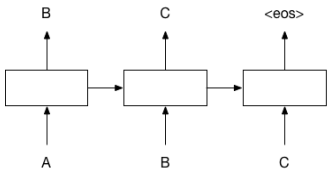

Figure 1: An RNN language model predicts a new token each time step, which is than the input for the next.

(Bahdanau et al., 2014). This system is the first user-tested translation aid utilizing NMT. Previous work such as Wuebker et al. (2016) and Knowles and Koehn (2016) have reported experiments with neural translation suggestion, however, they did not involve human translators.

In order to test the system we integrated it into Fairlingo[1]. It was trained and tested with the Dutch-English language pair, which is the most popular pair in Fairlingo.

2 Background

Computer aided translation (CAT) software products, e.g. SDL Trados Studio[2], utilize translation memories (TM). Translators can get suggestions through fuzzy matching of the source sentence against the memory. These suggestions are sentence-based and thus the translator's task is reduced to post-editing.

The Predictive Translation Memory (Green et al., 2014) uses a phrase-based MT system to create a smarter TM. It provides auto-complete similar to our system, however instead of completing words their system completes phrases.

Wuebker et al. (2016) compared such a phrase-based system to an NMT-based baseline system. They showed the phrase-based system was 10.6 to

[1]Fairlingo is a sharing economy platform for human translation. https://fairlingo.com

[2]https://sdl.com

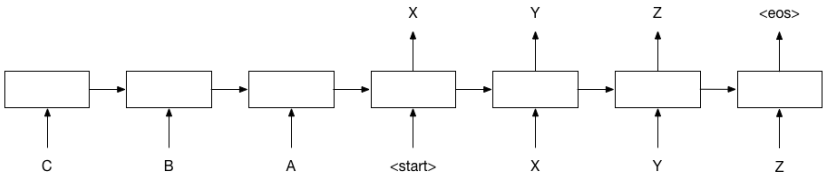

Figure 2: An NMT model which encodes "A B C" into a representation and decodes it into "X Y Z". Reversing the source sentence is an optimization by Sutskever et al. (2014). This figure was inspired by their paper as well.

31.3 times faster in making suggestions.

In contrast to phrase-based translation assistance, a NMT-based system is better able to recover from errors. Since the neural system makes word-by-word predictions, it can better adapt to unexpected user prefixes. Phrase-based systems, which are based on search graphs, may fail entirely if a prefix is not contained in the graph. Empirically, Knowles and Koehn (2016) showed that an NMT-based system yields higher word prediction accuracy (61.6% vs. 43.3%). We do not consider phrase-based translation assistance here.

3 Model description

The architectures of RNN models used for language modeling (LM) and NMT is especially suitable for a suggestion systems such as the one presented here. These models are entirely based around predicting the next token given a previously accepted token, as shown in Figure 1. However, instead of just predicting the $\arg\max$, i.e. the best token, one could also return a list of the n most likely tokens from the softmax output layer. These n-best tokens can then be used as direct suggestions or for further refinement using beam search. Another advantage of these models is their ability to predict a word given a partially translated target sentence, which is ideal for translation aids since one can feed the model with tokens from the partial translation to get suggestions for the next token.

We used an NMT model based on work by (Cho et al., 2014), and improvements by (Bahdanau et al., 2014). Figure 2 illustrates how a source sentence is encoded by the model into hidden states, i.e. a sentence representation, which is decoded into the target language. The model is trained jointly, even though it is composed of two separate parts: the encoder and the decoder. To allow the decoder to peek into the source sentence representation it uses an attention mechanism. For each word the decoder tries to predict, it takes soft-alignments to the encoder's hidden states into account.

	LM	NMT
Vocabulary	50,000	50,000
Layers	2	2
Units	1024 (GRU)	1024 (GRU)
Input	Context window[3]	Sentences
Attention	No	Yes
Bucketing	No	(5, 10, 20, 40)

Table 1: Summary of parameters and methods used in the models.

To rigorously test the efficiency of the NMT system in prediction, we also included a neural language model as a baseline prediction system. The implementation is based on the work by (Zaremba et al., 2014). Similar systems can often be found in smartphone keyboards where they yield good prediction performance. Table 1 shows the details of both models.

3.1 Training

One of the disadvantages of using NMT is that it requires a large amount of training data. Our models were trained on 20 million Dutch - English sentence pairs from the Open Subtitles 2016 parallel corpus (Lison and Tiedemann, 2016). We used a vocabulary of 50,000 tokens; a special unknown token was assigned to infrequent words.

The models were implemented using Tensorflow (Abadi et al., 2016) and were trained using the parameters shown in Table 1. Training of each model ran for 48 hours on a single *GeForce GTX 980 Ti* GPU.

4 System description

The Tensorflow library has well documented Python bindings, which makes it possible to integrate the model into a web application stack like Fairlingo's. A client-side application served the suggestions over a WebSocket, through which the

[3]The LM is trained using backprop through time. The size of the window is the size of the steps taken by that algorithm.

Figure 3: The interface the translators used to accept suggestions from the model.

model returns the 500-best list for next word suggestions when the previous word is completed. The translators can filter down the options by typing some characters. They can also accept one of the 8 visible suggestions immediately by selecting it using the arrow keys and pressing enter. Figure 3 shows how the suggestion system is presented to the end user.

5 User study

We asked four professional translators and two inexperienced translators to translate six documents in different settings. Each participant was asked to translate two sets of three documents, where in each set they translated once without aid, once with the LM aid and once with the NMT aid. For one set they were allowed to use all external tools they liked, and for the other set we asked them to refrain from using any extra tools (such as online dictionaries). Each translator was presented with a different document for each setting, however, all translators worked on the same documents. This ensures comparability of their translation performances. After the translator finished we asked them several questions about their experience with the translation aid.

The client-side application also measured each keystroke the translator entered. With keystroke information we can infer which words where accepted from the suggestions and which words were typed by the translator. Afterwards all translations were scored on quality through pairwise ranking on Crowdflower[4]. We collected 8500 judgements in total.

6 Evaluation

We are interested in finding out how neural translation assistance differentially affects two measures on a translator's output: translation time and translation quality. Both are extremely important when translation assistance systems are used in practice.

[4]Crowdflower is a platform for crowd sourcing tasks such as pairwise ranking. `https://crowdflower.com`

effects	mean (μ)	2.5%	97.5%
intercept	9.08	8.2	9.96
#source words	0.1	0.07	0.12
external tools	0.0	-0.28	0.3
professional	-0.25	-1.51	1.20
NMT probability	0.23	-0.09	0.55
LM probability	0.33	0.01	0.66
NMT suggestions used	0.08	-0.27	0.46
LM suggestions used	-0.06	-0.58	0.38
standard deviations (σ)			
global	0.52	0.48	0.57
ind. intercept	0.43	0.02	1.69
#source word	0.02	0.00	0.05
external tools	0.21	0.02	0.67
professional	0.27	0.01	1.06
NMT probability	0.17	0.01	0.57
LM probability	0.26	0.02	0.76
NMT suggestions used	0.19	0.02	0.65
LM suggestions used	0.35	0.02	1.10

Table 2: Means of coefficients and standard deviations (SD) with 95% credible intervals. The global SD is the SD not captured by the predictors, all other SDs are for individual-level adjustments to the intercept and coefficients. A higher SD means that the effect of the corresponding predictor is more variable across translators.

	LM	NMT
Words	7.85%	18.48%
Characters	7.87%	22.59%

Table 3: Percentage of word and characters auto-completed by accepting a suggestion.

6.1 Quantitative results

We use a hierarchical Bayesian linear regression model (Gelman and Hill, 2007) to estimate effect sizes on translation time. Besides estimating the effects of the translation aids on "the average translator" it also gives us an estimate of the variability of these effects between translators. The fact that our model is Bayesian also means that we make rather conservative estimates.[5] A further advantage of the Bayesian methodology is that our results can easily be integrated with results from future studies of the same kind.

We perform regression on the log-transformed time it took to translate a sentence. The log-

[5]Many of the effects reported here are statistically significant at $\alpha = 0.5$. Looking at the distribution of these effects is more informative, however, especially with regard to future research that may use our estimates as priors.

Suggestion sytem		Disabled	Enabled
NMT	1st suggestion	50.44%	51.97%
	in 8-best list	81.27%	82.46%
	in 500-best list	96.23%	97.31%
LM	1st suggestion	17.77%	16.89%
	in 8-best list	44.67%	43.96%
	in 500-best list	82.30%	82.39%

Table 4: Matches between the word suggested by the system and the word entered by the translator. The 8-best suggestions where shown directly, the 500-best suggestions could be reached by typing some characters.

transformation makes our data approximately normal. In Table 2 we see that the introduction of translation aids slows translators down somewhat. This is in line with previous findings (Green et al., 2014). When the LM aid is used to complete words, it enhances translation speed as compared to the NMT aid. We conjecture that this is so because the neural translation aid offers more plausible suggestions and thus it takes the translator longer too choose from those. Reducing the size of the k-best list shown to the translators may thus be beneficial. Unsurprisingly, professional translators are faster on average than non-professional ones.

On the pairwise ranking task, we found that translations produced with different translation aids were judged superior an equal amount of times. This indicates that while the aids may support translators, they cannot yet bridge the gap in skill that often exists between individual translators.

For translation quality, we also assessed how often the translation aid predicted a word that the translator did indeed use. Table 3 shows that the NMT aid gives the translators more suggestions which they actually accept, as opposed to the LM system. The ratio of words to characters is also higher which indicates that an NMT system is used to complete longer words. Furthermore, in Table 4, we can see the guiding effect that the translation aid has on the translators. It shows that the suggestions made by the NMT system often matches the words that the translators actually typed. This is the case even when the translators do not make use of the auto-complete option.

6.2 Qualitative results

CAT tools mostly use post-editing features, and translators in general find post-editing an extremely boring, tedious and unrewarding chore (Church and Hovy, 1993; Koehn, 2009). When asked about CAT tools, the participants of the user-study confirmed this. Only one of the participating professional translators reported that he actually uses a CAT tool. The majority of the translators preferred working in a regular word processor.

Five of the six translators found the NMT-based suggestions to be unobtrusive and helpful in their task, only two judged the LM system similarly.

Several translators remarked that one of the things they disliked about our system and translation aids in general is that most are sentence focused. Text is not displayed in its proper context, i.e. in a paragraph. Furthermore, one of the translators remarked that suggestions on the word level made it hard for them to focus on the entire sentence.

Our assignment contained various compound word on the source side whose English translations are multi-word expressions, e.g. ”arbeidsongeschiktheidsverzekering” (work-related disability insurance). All translators told us they would have liked the system to give good suggestions for these words, since these were generally the words that they found difficult to translate. This might be a good indication that suggestion systems could look to sub-word or character based models (Chung et al., 2016) for languages with compounds.

7 Conclusion

We have shown that neural translation aids are a viable option for interactive translation assistance. NMT-based systems tend to deliver faster translations than LM-based systems and their effect on individual translators is less variable. However, translators need to get used to these systems. We conjecture that by reducing the size of the k-best list translation speed may be further improved since the translators have to look at fewer competing options.

In the future, we aim to extend our study with more translators to get more representative estimates of the effects that influence translator performance. After consulting with our translators, we also plan to include multi-word suggestions in our system.

References

Martın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al.

2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Ouais Alsharif, Tom Ouyang, Françoise Beaufays, Shumin Zhai, Thomas Breuel, and Johan Schalkwyk. 2015. Long short term memory neural network for keyboard gesture decoding. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2076–2080. IEEE.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

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

Kenneth W Church and Eduard H Hovy. 1993. Good applications for crummy machine translation. *Machine Translation*, 8(4):239–258.

Andrew Gelman and Jennifer Hill. 2007. *Data analysis using regression and multilevel/hierarchical models*. Cambridge University Press.

Spence Green, Jason Chuang, Jeffrey Heer, and Christopher D Manning. 2014. Predictive translation memory: A mixed-initiative system for human language translation. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pages 177–187. ACM.

Rebecca Knowles and Philipp Koehn. 2016. Neural interactive translation prediction. *AMTA 2016, Vol.*, page 107.

Philipp Koehn. 2009. A process study of computer-aided translation. *Machine Translation*, 23(4):241–263.

Pierre Lison and Jörg Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*.

Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 553–562. ACM.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Joern Wuebker, Spence Green, John DeNero, Saša Hasan, and Minh-Thang Luong. 2016. Models and inference for prefix-constrained machine translation. *54th ACL*, 1:66–75.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.