Character-Level Transformer-Based Neural Machine Translation

Nikolay Banar
CliPS, University of Antwerp
ACDC, University of Antwerp
Antwerp, Belgium
nicolae.banari@uantwerpen.be

Walter Daelemans
CliPS, University of Antwerp
Antwerp, Belgium
walter.daelemans@uantwerpen.be

Mike Kestemont
CliPS, University of Antwerp
ACDC, University of Antwerp
Antwerp, Belgium
mike.kestemont@uantwerpen.be

ABSTRACT

Neural machine translation (NMT) is nowadays commonly applied at the subword level, using byte-pair encoding. A promising alternative approach focuses on character-level translation, which simplifies processing pipelines in NMT considerably. This approach, however, must consider relatively longer sequences, rendering the training process prohibitively expensive. In this paper, we discuss a Transformer-based approach, that we compare, both in speed and in quality to the Transformer at subword and character levels, as well as previously developed character-level models. We evaluate our models on 4 language pairs from WMT'15: DE-EN, CS-EN, FI-EN and RU-EN. The proposed architecture can be trained on a single GPU and is 34% faster than the character-level Transformer; still, the obtained results are at least on par with it. In addition, our proposed model outperforms the subword-level model in FI-EN and shows close results in CS-EN. To stimulate further research in this area and close the gap with subword-level NMT, we make all our code and models publicly available.

CCS Concepts

•**Computing methodologies** → **Machine translation;**

Keywords

Natural Language Processing; Neural Machine Translation; Character-Level Translation

1. INTRODUCTION

Sequence-to-sequence models are nowadays a mainstream approach in Neural Machine Translation (NMT). Such models are typically applied at the subword level based on byte-pair encoding (BPE), originally proposed by Sennrich et al. [26]. This algorithm mitigates the problem of rare and out-of-vocabulary words that present a significant issue for word-level models. BPE builds a vocabulary of the most frequent subword units of different lengths, starting from a single character. Then, the input sentence is divided into a sequence of the longest possible subword fragments matching the constructed vocabulary. This approach is appealing because of its strong empirical results and computational efficiency. However, the segmentation is language- and corpus-dependent and, hence, requires considerable hyperparameter tuning. The problem of finding an optimal subword segmentation is especially challenging for multilingual and zero-short translation [12].

Another recent direction in NMT focuses on character-level translation. This approach is conceptually attractive because it can help mitigate the previously mentioned shortcomings of subword-level models. Character-level models do not rely on an explicit segmentation of the input sentence (be it rule-based or statistical) and resort to plain characters as a sentence's basic units. As such, models are implicitly enforced to learn the inner structure of complex words. Hence, such models are more robust in the face of out-of-vocabulary words and in translating noisy and out-of-domain text. In comparison to subword-level models, they should be able to model more accurately rare morphological variants of words [7, 17, 10]. In addition, character-level models may work better in some fine-tuning scenarios, where the amount of available data is challengingly small [2].

In spite of its conceptual elegance, the character-level approach also presents considerable challenges, that help explain why this approach did not receive much attention yet. Character sequences are significantly longer and, consequently, more challenging to model. Moreover, the level of semantics in character-level representation becomes even more abstract and, hence, larger models with a highly non-linear mapping function are required. Finally, the training and decoding time for such models is much longer. However, some of these issues can be tackled through resorting to new NMT architectures. Lee et al. [17] have shown that is possible to train a character-level model, within a reasonable time span, by reducing the length of the source representation. We utilize this publicly available model, henceforth: CharRNN, as a baseline in our experiments.

We base our work on the well-known Transformer architecture [29], which has shown state-of-the-art performance on several language pairs in NMT. The model is intrinsically very attractive for the character level due to the high training speed it enables and its strong modelling capacity with respect to longer-range dependencies. The Transformer relies on self-attention and does not include any recurrence

NLPIR 2020, December 18–20, 2020, Seoul, Republic of Korea

ISBN 978-1-4503-7760-7/20/06…$15.00

DOI: https://doi.org/10.1145/3443279.3443310

in training. Therefore, the Transformer can be fully parallelized during training, leading to considerable speed-ups in comparison to recurrent networks.
We aim to stimulate further research in this direction, by demonstrating the computational feasibility of training fast character-level models, even on a single GPU. Below, we propose a new variant (CharTransformer) of a publicly available, Transformer-based network and apply it at the character level. Our models applies the same source length reduction technique as Lee et al. [17] and introduces a six-layer Transformer at the encoder and decoder sides instead of recurrent layers as in CharRNN, making our network fully parallelizable. The main contribution of the paper is two-fold: (i) We demonstrate the feasibility of training high-quality and fast character-level translation models, even on a single GPU; (ii) we propose a novel character-level Transformer-based architecture that is at least as accurate as the Transformer, yet is up to 34% faster.

2. RELATED WORK

In this section, we survey recent work in the field of character-level NMT that is directly relevant to the present paper. Costa-jussà and Fonollosa [8] utilized a convolutional network to extract local dependencies from character embeddings and, downstream, applied a Highway network [27] to construct segmented embeddings. This model showed promising results but, crucially, still relied on a word-level segmentation at the decoder and encoder sides. Ling et al. [18] assembled word embeddings from character embeddings via bidirectional long short-term memory units (LSTM) [11]. The model decoded the target words character-by-character and outperformed a comparable word-based baseline. However, the training time was substantially longer and, still, explicit segmentation was required.
Luong and Manning [19] used character-level information to mitigate out-of-vocabulary issues in a word-based model. Additionally, they compared a fully character-level model with a word-level baseline. Notwithstanding comparable results, the fully character-level model was significantly slower. Chung et al. [7] compared character-level and subword-level decoders, while the encoder still worked at the subword level. Their experiments demonstrated that the character-level decoder could outperform the subword-level one.
Lee et al. [17] were the first to propose a fully character-level model that came with computational requirements comparable to those of subword-level models. At the encoder side, they efficiently reduced the length of the input sequences via the use of a convolutional layer, a max-pooling layer and a stack of Highway layers. On top of the encoder, they used bidirectional gated recurrent units (GRU) [6]. In this paper too, the character-level NMT model was able to outperform the subword-level baseline. Finally, and in the same spirit, Cherry et al. [4] showed that standard character-level models of sufficient depth are able to outperforms comparable subword-level models. However, they utilized a prohibitively expensive training regime with 16 GPUs (training times were not explicitly reported for each network) and did not make their models publicly available. Hence, we do not consider these models below and restrict ourselves to publicly available implementations. Gupta et al. [10] demonstrated that the character-level Transformer is competitive to the subword-level Transformer, but does not outperform it.
Here, we take inspiration from Chen et al. [3], who investigated different NMT architectures, including hybrid models with Transformers. They demonstrated the superiority of the Transformer encoder over the recurrent encoder at the subword level. We hypothesize that the CharRNN model may be easily improved by incorporating the Transformer approach, instead of the more conventional, recurrent layers. In addition, the architecture can be sped up at the training phase by using the Transformer decoder (as in CharTransformer). Our work is therefore the first to assess the effectiveness and efficiency of CharTransformer.

3. BACKGROUND

Table 1: Encoder and decoder parameters of the investigated models. At the encoder side, the models utilize 200 filters of width 1, 200 filters of width 2 etc. d_{ff} corresponds to the inner-layer has dimensionality. d_m corresponds to the dimensionality of input and output. d_k, d_v correspond to the dimensionality of keys and values for attention heads, respectively.

Encoder		
Param.	**Transformer**	**CharTrans.**
Emb.	512	128
Conv. filters		200-200-250-250 300-300-300-300
Pool stride		5
Highway		2
Layers	6	
d_m, d_k, d_v	512	
Heads	8	
d_{ff}	2048	

Decoder		
Param.	**Transformer**	**CharTrans.**
Emb.	512	
Layers	6	
d_m, d_k, d_v	512	
Heads	8	
d_{ff}	2048	

In this section, we briefly discuss two of the commonly used architectures in NMT.

3.1 Recurrent Neural Networks

Recurrent models nowadays generally utilize GRU or LSTM memory cells, and follow the encoder-decoder paradigm. They consist of an encoder and an (attentional) decoder [1, 28, 20, 5]. The encoder processes a source sentence and constructs a continuous representation of it, which is sometimes considered a summarized meaning of the input sentence. The decoder generates the output sentence. These models are usually trained by minimizing the negative conditional log-likelihood of outputs given the corresponding source sentences and the previously observed target tokens.

3.1.1 Encoder

The encoder processes a source sentence step by step and the current state of the encoder depends on its previous

hidden state. A common practice is to apply bidirectional recurrent layers. A forward recurrent layer processes the input sequence from left to right and a backward recurrent layer processes it from right to left. Further, the outputs of the layers are concatenated in order to assemble the final source sentence representation.

3.1.2 Attentional Decoder

Depending on the specific architecture, the input of the decoder may include the previously generated token, its previous hidden states and the the context vector. The context vector is built by the attention mechanism. It searches parts of the source sentence that are relevant for each decoding time step. The context vector is calculated as a weighted sum of the source hidden states. Hence, the weights represent an importance of the input tokens given the current target token.

3.2 Transformer

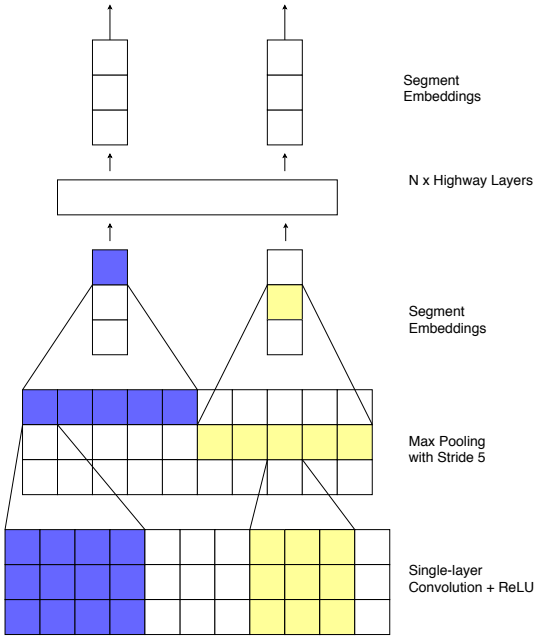

Figure 1: Scheme of the source length reduction technique.

The Transformer [29] model aims to overcome some of the issues induced by recurrent and convolutional sequence-to-sequence models. Compared to convolutional models, which have a limited receptive field, the Transformer utilizes self-attention networks. Thereby, the model is able to access all position of the previous layer. In addition, the Transformer does not have any recurrent connections at the training phase that allows to make training process fully parallel. These NMT models still rely on encoder-decoder scheme, which follows the same purpose as for recurrent networks. Transformers are commonly trained using the Noam decay schedule [24], also by minimizing the negative conditional log-likelihood.

3.2.1 Encoder

The encoder processes the full sequence simultaneously, as opposed to recurrent approaches. It starts with a positional encoding and processes the full sequence at once. As the Transformer contains no recurrence and no convolution, this step is required to provide information about the position of the tokens in the sequence. The encoder in each layer consists of 2 sub-layers: a self-attention network and a feed-forward neural network. In addition, a residual connection around each sub-layer is utilized. Downstream, layer normalization is applied. The encoder, because of its immediacy, is fully parallelizable in training and decoding phases.

3.2.2 Decoder

In comparison to the encoder, decoder layers have an additional self-attention network between 2 sub-layers that attend to the encoder. The decoder is fully parallelizable in the training phase. However, decoding is conducted step by step similarly to recurrent networks.

4. MACHINE TRANSLATION MODELS

In this work, we compare three character-level and one subword-level NMT systems. First, we report results for the character-level model proposed by Lee et al. [17] and use it as a baseline (CharRNN). In this model, the decoder consists of two unidirectional GRU layers and the attention score is computed by a single-layer feedforward network. The encoder part implements an efficient source length reduction technique (detailed below), and adds a single-layer, bidirectional GRU on top. Second, we train a character-level Transformer and a subword-level Transformer [29] without any architectural modifications. And finally, we apply the source length reduction technique to the Transformer and build CharTransformer. We implemented this model in PyTorch [23], inside the OpenNMT-py framework [15]. Further information about the parameters of the encoders and the decoders of the Transformer and CharTransformer are summarized in Table 1. Layer sizes of the models are kept maximally comparable. Below, we highlight the important details of the models.

4.1 Source Length Reduction

As a recurrent baseline model, we use the model proposed by Lee et al. [17]. The encoder employs one-dimensional convolutions, following with max-pooling layers and a Highway network, in order to reduce the substantial length (up to 450 characters) of the input sentence by a factor of 5 and efficiently construct representation of local features. We briefly highlight the main properties of the source length reduction technique below, which is schematically depicted in Figure 1.

4.1.1 Embedding layer

The embedding layer takes the form of a lookup table, which maps a sequence of source tokens to a sequence of embeddings in order to build a continuous representation of each token.

4.1.2 Convolutions

One-dimensional convolutional filters (with padding) are applied to the sequence of the input embeddings produced by the embedding layer. Filter widths range from 1 to 8, which allows to construct representation of n-grams up to 8 characters. Downstream, the outputs of the convolutional filters are stacked and the rectified linear activation is applied.

4.1.3 Max pooling

Conventional max pooling is applied to non-overlapping parts of the convolutional layer output. Thus, the layer reduces the length of the source representation and constructs segment embeddings, containing the most salient features of the source sub-sequences.

4.1.4 Highway layers

The Highway network is introduced after the convolutional part of the encoder. Highway layers [27] have been shown to improve the quality of character-level models [13].

4.2 CharTransformer Encoder

In the CharTransformer encoder, we implement the source length reduction technique from Lee et al. [17] (Figure 1) and inherit the following layers from the baseline: the embedding layer, the convolution layer, the max pooling, the Highway network. On the top of the encoder, we employ a six-layer Transformer.

5. EXPERIMENTAL SETTINGS

5.1 Datasets and Preprocessing

We applied the NMT models to the four language pairs from WMT'15: DE-EN, CS-EN, FI-EN and RU-EN. We obtained the datasets[1] already preprocessed by Lee et al. [17], using a script from Moses [16]. Although this step is not strictly required for character-level translation, we kept it for the sake of comparison. In addition, we created a tokenized dataset, using another reference routine [26], with 20,000 BPE operations for each of the source and target corpora. We allowed a vocabulary size of 300 tokens for the character-level translation and 20k−24k tokens for the subword-level models. We limit the length of sentences to 450 characters or 50 subword tokens. For the FI-EN language pair, we utilized newsdev-2015 as a development set and newstest-2015 as a test set. For other language pairs, we used newstest-2013 as a development set and the combination of newstest-2014 and newstest-2015 as test sets.

5.2 Metrics

Notwithstanding its reliability, human assessment in machine translation is expensive and slow to obtain. In NMT, a number of automated metrics have therefore been proposed to measure the performance of models. Generally speaking, these measure the quality of a system's output by comparing it to human judgments. Recently, character-level metrics demonstrated the best performance among the non-trainable metrics in the field [21]. Therefore, we utilized not only the popular metric BLEU [22], but also CHARACTER [30] and CHRF [25].

5.3 Training Details

We mostly followed the settings recommended by the OpenNMT-py framework[2]. The models were trained by minimizing the negative conditional log-likelihood using the Adam optimizer [14] with an initial learning rate of 2 and the Noam decay schedule [24]. The models were initialized using the method proposed by Glorot and Bengio [9]. We did not change any settings for the subword-level models. Below, the parameters that we altered for the character-level models are explicitly listed. As character tokens contain less information compared to subwords, we utilized a larger batch size of 6144 tokens and an accumulation count of 4, to get a more faithful gradient approximation. Additionally, we set dropout to 0 to make the models converge faster. We used -max_generator_batches with default parameters. We trained the models for 100,000 updates. Each model was trained on a single GeForce GTX 1080 Ti with 11 GB of memory.

[1]https://github.com/nyu-dl/dl4mt-c2c
[2]https://opennmt.net/OpenNMT-py/FAQ.html

5.4 Encoding Details

We slightly altered the implementation of the original source length reduction used by Lee et al. [17] in CharRNN to reduce the memory consumption of the model. Highway layers significantly improve the performance of character-level language models based on convolutional networks. Even though, the Highway layers significantly improve the performance of convolution based character-level language models, Kim et al. [13] demonstrated that they saturate in performance after 2 layers. Therefore, we utilized only 2 (instead of the original 4) layers in CharTransformer to reduce the complexity of the models under consideration.

5.5 Decoding Details

In the decoding part, we utilized beam search with beam size of 20 for character-level models and beam size of 5 for subword-level models.

6. RESULTS AND DISCUSSION

6.1 Quantitative Analysis

6.1.1 Instability of metrics

Interestingly, we can observe a high variation in metrics (see Table 2). However, it is expected due to different degree of correlation between metrics and human scores. If we rely solely on highly popular BLEU conclusions may be misleading as it is not the best metric for three out of four language pairs (see Table 4). From Table 2, we can see that improvement of 1 BLEU point does not necessary lead to improvements in other metrics. Hence, we make our conclusions based on least two metrics out of three where it is possible.

6.1.2 RNN vs. Transformer

Lee et al. [17] reported a training time for CharRNN of approximately 2 weeks on a single GPU. However, we can not directly compare training time of CharRNN to our character-level models due to usage of different frameworks, GPUs, batch sizes and depth of models. From Table 3, we can observe that it takes roughly 38 and 25 hours to train the character-level Transformer and CharTransformer respectively. In addition, the character-level Transformer and CharTransformer show better results for all language pairs (see Table 2). Hence, we train our deeper character-level models substantially faster and outperform previously obtained results by a large margin. We conclude that Transformer applied at the character level and CharTransformer are better than CharRNN.

6.1.3 Character-level Transformer vs. CharTransformer

Table 2: Results of the models on 4 language pairs. The best performing models are shown in bold. Results for CharRNN are obtained from Lee et al. [17].

Lang.	Model	Seg.	Test1			Test2		
			BLEU↑	C-TER↓	CHRF↑	BLEU↑	C-TER↓	CHRF↑
DE-EN	CharRNN	char	25.77	NA	NA	25.83	NA	NA
	Transformer	char	28.32	47.41	53.14	28.70	45.44	53.08
	CharTransformer	char	28.63	46.54	53.70	28.08	**45.16**	53.18
	Transformer	bpe	**29.72**	**46.35**	**54.26**	**29.76**	45.36	**54.11**
CS-EN	CharRNN	char	24.08	NA	NA	22.46	NA	NA
	Transformer	char	24.77	48.13	50.91	23.51	51.34	48.20
	CharTransformer	char	26.89	**45.40**	53.66	25.24	**49.44**	50.47
	Transformer	bpe	**28.41**	45.62	**54.02**	**26.14**	49.92	**50.56**
FI-EN	CharRNN	char	NA	NA	NA	13.10	NA	NA
	Transformer	char	NA	NA	NA	**18.72**	**55.95**	**44.97**
	CharTransformer	char	NA	NA	NA	17.52	57.70	43.46
	Transformer	bpe	NA	NA	NA	17.35	58.21	42.90
RU-EN	CharRNN	char	26.80	NA	NA	22.73	NA	NA
	Transformer	char	30.87	**42.55**	**56.80**	26.99	**46.17**	52.96
	CharTransformer	char	30.31	42.78	56.35	26.19	46.72	52.21
	Transformer	bpe	**31.39**	43.21	56.75	**28.01**	46.40	**53.41**

Table 3: Speed comparison for the character-level models. The second column shows the time of one update in seconds. The third column reports the total training time in hours. The last column shows speed difference in percents. The models make one update after processing four batches.

Model	Speed	Overall	Percent
Trans.	1.362	37.71	100
CharTrans.	0.894	24.76	66

Table 4: WMT15 system-level correlations of automatic evaluation metrics and the official human scores [30]. The best results are in bold.

Metric	FI-EN	DE-EN	CS-EN	RU-EN
C-TER	0.888	**0.972**	0.960	0.884
CHRF	0.903	0.956	**0.968**	**0.898**
BLEU	**0.929**	0.865	0.957	0.851

According to Table 2, Transformer applied at character level is the best performer in FI-EN and RU-EN. CharTransformer shows better results in DE-EN and CS-EN. In the experiments, we do not observe superiority of CharTransformer in results over Transformer. However, CharTransformer is 34% faster. We conclude that CharTransformer is promising and worth further investigation.

6.1.4 Character- vs. subword-level

From Table 2, we can observe that character-level models in some cases outperform subword-level models. CharTransformer and character-level Transformer outperform subword-level Transformer in FI-EN. In addition, character-level Transformer shows comparable results in RU-EN and CharTransformer is slightly worse in CS-EN than subword-level Transformer. The subword-level model is convincingly the best only in DE-EN. Similarly to Gupta et al. [10], we observe that the character-level models are competitive to the subword-level models, but do not outperform them. It shows that these models are promising and should get more attention.

6.2 Qualitative Analysis

We have performed a qualitative inspection of 100 randomly sampled sentences from newstest-2014 of the Russian-English language pair for the four models compared (CharRNN, subword-level and character-level Transformer, and CharTransformer). We selected this language pair because of the relatively large typological distance between both languages, as well as the challenging transliteration issues that might arise from the mapping of two alphabets. Overall, CharRNN displays a clear inferiority to the Transformer architectures. The quality of CharTransformer is indeed slightly lower than the Transformers (in accordance with the quantitative results), but not much. Noteworthy are the following, persisting error categories (referencing examples a–d drawn from Table 5)

6.2.1 Entities and transliteration

Named entities, especially proper nouns, are a classic hindrance in NMT, especially when source and target language use a different alphabet. All systems suffer from artifacts in this area, but CharRNN most heavily. In many cases, systems propose entirely different transliterations of the proper nouns in the source language (a).

6.2.2 Length-related artifacts

CharRNN translations often feature the unnecessary repetitions of chunks ('flooding'), as well as incomplete words (b). Likewise, CharRNN often produces incorrect syntactic constructions which is rare with the other architectures. Overall, the Transformers yield slightly more concise translations than the CharTransformer (121.58±59.92 (bpe) vs. 125.18±64.80 (char) vs. 126.11±64.94 characters on average) (d), which might be related to the settings of the beam search.

Table 5: Examples of translation from CharRNN, Transformer and CharTransformer, illustrating the main error types, observed in a random sample of 100 sentences for the Russian to English language pair.

(a) Named Entities and transliteration (Russian→English)

transliteration	Ostaviv ej golosovoe soobshhenie 18 ijunja 2005-go , **Koulson** skazal : [...]
target	Leaving the voice message on June 18 , 2005 , **Caulsen** said : ' [...]
CharRNN	Having left her voicemail on 18 June 2005 , **Coleson** said , ' [...]
Transformer (char)	Having left her voicemail on 18 June 2005 , **Coleson** said , ' [...]
CharTransformer	Leaving her voice message on June 18 , 2005 , **Cowlson** said , ' [...]
Transformer (bpe)	Leaving her a voice message on 18 June 2005 , **Colson** said , ' [...]

(b) Flooding of chunks and incomplete words (Russian→English)

transliteration	Sirija unichtozhila oborudovanie dlja himoruzhija
target	Syria destroyed equipment for chemical weapons
CharRNN	Syria destroyed the **equipment** for the **equipment** for **chemothera**
Transformer (char)	Syria has destroyed chemo-weapons equipment
CharTransformer	Syria Destroyed Chemical Equipment
Transformer (bpe)	Syria Destructed Chemical Weapons

(c) Fixed expressions (Russian→English)

transliteration	V Kineshme i rajone dvoe muzhchin pokonchili zhizn' samoubijstvom
target	In Kineshma and environs two men have committed suicide
CharRNN	In Kineshma and the area **of** two men **committed suicide behavior**
Transformer (char)	In Kineshma and the region , two men have committed suicide .
CharTransformer	In Kineshma and the region , two men have ended their lives **of suicide**
Transformer (bpe)	In Kineshma and environs two men have committed suicide

(d) Conciseness of Transformer (Russian→English)

transliteration	Ko vremeni podvedenija itogov tendera byla opredelena arhitekturnaja koncepcija ajerovokzal'nogo kompleksa ' Juzhnyj ' , kotoruju razrabotala britanskaja kompanija Twelve Architects
target	By the time the tender results were tallied , the architectural concept of the ' Yuzhniy ' air terminal complex , which was developed by the British company Twelve Architects , had been determined .
CharRNN	By the time the tender 's results were defined an architectural concept of the ' South ' architecture complex , which was developed by the British company Twelve Architects .
Transformer (char)	By the time of summing up the results of the tender the architectural concept of the Yuzhny terminal complex was developed by Twelve Architects .
CharTransformer	By the time of the summing up of the tender , the architectural concept of the ' South ' terminal complex developed by the British company Twelve Architects was identified .
Transformer (bpe)	By the time the tender results were summed up the architectural concept of the Yuzhny airport terminal complex developed by British company Twelve Architects.

6.2.3 *Fixed expressions*

In comparison to the Transformer architectures, CharRNN sometimes struggles to translate figurative language use and idiomatic expressions. The same is true for the CharTransformer, but to a lesser extent (c).

6.2.4 *Overall quality*

We conclude that CharRNN is relatively less capable of modelling longer-range sequences at the character level. To the human eye, and however small the sample size, the differences between the Transformers and CharTransformer are limited, although the Transformers generally yields minimalist translations, that are of a slightly higher quality.

7. CONCLUSION AND FUTURE WORK

In this work, we applied Transformer from OpenNMT-py at character level and proposed a new character-level Transformer-based NMT architecture, CharTransformer. We evaluated it on four languages from WMT'15 corpora and compared these models to the character-level architecture previously proposed by Lee et al. [17]. We showed that character-level Transformer and CharTransformer outperform this model in all tasks. We demonstrated that character-level translation does not require weeks of training and expensive multi GPU training scheme anymore to obtain strong results. In addition, we showed that CharTransformer performs comparably with character-level Transformer and is 34% faster. CharTransformer outperforms the subword-level model in FI-EN and shows competitive results in CS-EN. We conclude that both models are promising for character-level translation and can stimulate further research in this field.

We provide the following repository[3] that contains the source code of the implemented models and the corresponding weights. In future research, we would like to investigate multilingual character-level translation with Transformer and Char-Transformer. In addition, we will research different properties of these models. Finally, we should emphasize that our results that we might close the gap between character-level and subword-level NMT in a very near future.

8. REFERENCES

[1] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations, ICLR 2015*, 2015.

[2] N. Banar, K. Lasaracina, W. Daelemans, and M. Kestemont. Transfer learning for digital heritage collections: Comparing neural machine translation at the subword-level and character-level. In *Proceedings of the 12th International Conference on Agents and Artificial Intelligence - Volume 1: ARTIDIGH,*, pages 522–529. INSTICC, SciTePress, 2020.

[3] M. X. Chen, O. Firat, A. Bapna, M. Johnson, W. Macherey, G. Foster, L. Jones, M. Schuster, N. Shazeer, N. Parmar, et al. The best of both worlds: Combining recent advances in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–86, 2018.

[4] C. Cherry, G. Foster, A. Bapna, O. Firat, and W. Macherey. Revisiting character-based neural machine translation with capacity and compression. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4295–4305, 2018.

[5] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, 2014.

[6] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014.

[7] J. Chung, K. Cho, and Y. Bengio. A character-level decoder without explicit segmentation for neural machine translation. In *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016*, pages 1693–1703. Association for Computational Linguistics (ACL), 2016.

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

[9] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[10] R. Gupta, L. Besacier, M. Dymetman, and M. Gallé. Character-based nmt with transformer. *arXiv preprint arXiv:1911.04997*, 2019.

[11] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[12] M. Johnson, M. Schuster, Q. V. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. Viégas, M. Wattenberg, G. Corrado, et al. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351, 2017.

[13] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush. Character-aware neural language models. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

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

[15] G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. M. Rush. OpenNMT: Open-source toolkit for neural machine translation. In *Proc. ACL*, 2017.

[16] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics, 2007.

[17] J. Lee, K. Cho, and T. Hofmann. Fully character-level neural machine translation without explicit segmentation. *Transactions of the Association for Computational Linguistics*, 5:365–378, 2017.

[18] W. Ling, I. Trancoso, C. Dyer, and A. W. Black. Character-based neural machine translation. *arXiv preprint arXiv:1511.04586*, 2015.

[19] M.-T. Luong and C. D. Manning. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1054–1063, 2016.

[20] M.-T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015.

[21] Q. Ma, O. Bojar, and Y. Graham. Results of the wmt18 metrics shared task: Both characters and embeddings achieve good performance. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 671–688, 2018.

[22] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[23] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein,

[3] http://doi.org/10.5281/zenodo.3988362

L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8026–8037, 2019.

[24] M. Popel and O. Bojar. Training tips for the transformer model. *The Prague Bulletin of Mathematical Linguistics*, 110(1):43–70, 2018.

[25] M. Popović. chrf: character n-gram f-score for automatic mt evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, 2015.

[26] R. Sennrich, B. Haddow, and A. Birch. 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, 2016.

[27] R. K. Srivastava, K. Greff, and J. Schmidhuber. Training very deep networks. In *Advances in neural information processing systems*, pages 2377–2385, 2015.

[28] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[30] W. Wang, J.-T. Peter, H. Rosendahl, and H. Ney. Character: Translation edit rate on character level. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 505–510, 2016.