Hybrid Word-Character Neural Machine Translation for Modern Standard Arabic

Pamela Toman
Department of Computer Science
Stanford University
Stanford, CA 94305
ptoman@stanford.edu

Sigtryggur Kjartansson
Department of Computer Science
Stanford University
Stanford, CA 94305
sigkj@stanford.edu

Abstract

Traditional neural machine translation architectures use a word-level approach that assumes all important words have enumerable and relatively frequent surface forms. This assumption is invalid for the large number of non-analytic languages that form new words on the basis of complex morphological processes. Contributing to the quest of finding a universalist architecture that performs well for all language pairs, Luong and Manning recently suggested a hybrid model that backs off to character-level representations when word-level representations are unavailable. We reimplement Luong and Manning's hybrid model in TensorFlow and apply it to Arabic machine translation. With this architecture, we are able to repeatedly saturate our models and achieve a best BLEU score of 42.05, which approaches state-of-the-art.

1 Introduction

Traditional neural machine translation and statistical translation approaches assume that the basic unit of meaning for translation is the word. However, this is a false assumption for less analytic languages, and it leads to a data sparsity problem for languages with rich morphologies. In particular, in contrast to analytic languages like English, the surface form of a word in a synthetic language may be different from any words observed during training, yet recoverable if understood in terms of its observed morphological parts.

Although morphological complexity is common in languages, techniques for addressing it have been under-studied relative to its real-world frequency. We seek to improve performance of neural machine translation in languages with rich morphologies. To do so, our approach unites three models. The core model is a word-level sequence-to-sequence with global bilinear attention. Two additional character models support the word-level model and are trained simultaneously to it. In the source language, we develop meaningful dense word embeddings on-the-fly for unknown tokens, and in the target language, when the word model generates an unknown token, we replace that token with a generated character-level sequence. This approach is based on Luong and Manning 2016 [1].

With a hybrid word-character approach that balances the computational efficiency of a word model with the arbitrary representations possible in a character model, we achieve a BLEU score of 42.05 on Arabic to English machine translation. Given that we were able to saturate multiple increasingly large models, we expect even further gains are possible with this architecture.

2 Background

Neural machine translation (NMT) directly models the probability of target language sentence $(y_1, y_2, \ldots, y_n)$ given a source language sequence $(x_1, x_2, \ldots, x_m)$ using recurrent neural net-

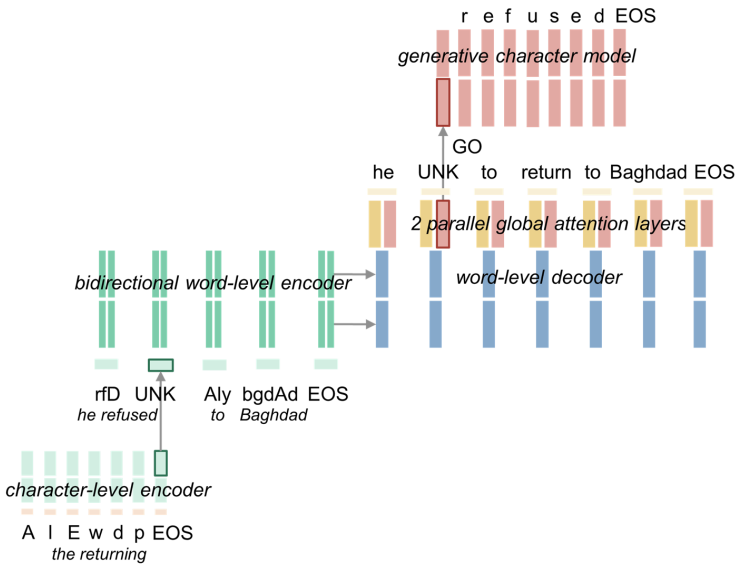

Figure 1: Hybrid architecture.

works. Basic NMT has a word-level encoder, which computes a representation of a sentence $\dot{h}$, and a word-level decoder, which uses the representation of the sentence $\dot{h}$ and the current decoder hidden state h_t to generate one word at a time. The model maximizes the overall sentence probability $\log p(y|x)$:

$$\log p(y|x) = \sum_{j=1}^{n} \log p(y_j|y_{<j}, h_t, \dot{h}) \tag{1}$$

Neural models differ in their unit of analysis and the encoder representation $\dot{h}$.

Traditionally, neural machine translation models operate with word as their level of analysis, and they replace all words x_i and y_j below a threshold frequency with the unknown token UNK. Recently, however, researchers have begun to focus on performing translation with sub-word structure. Although working purely at the character-level can be challenging because of the size of the unrolled RNN, a number of researchers have been making progress in this area:

- Ling et al. [2] introduce a straightforward character-based RNN for NMT.
- Kim et al. [3] and Lee et al. [4] build character-based CNNs for NMT.
- Chung, Cho and Bengio [5] perform neural machine translation at the character level using a bi-scale RNN with “fast" and “slow" layers.

Alternative approaches that leverage the core computational speed of a word-based model with additional information for unknown words include:

- Sennrich et al. [6], who use character n-grams to encode rare words through subword units.
- Luong et al. [7], who suggest a two-pass system to translate target UNK words by revisiting the original dataset.

We follow Luong and Manning [1], who introduce a hybrid neural machine translation model that backs off to character-level modeling when word-level representations are unavailable.

3 Approach

We unite three models to create a hybrid word-character model, as illustrated in Figure 1. The core of the model is word-level translation, augmented with character-level models where UNK tokens appear. The source language character-level encoder creates word embeddings for UNK words, and the target language character-level decoder generates text for UNK words predicted by the word-level

model. With this approach, we are able to balance the efficiency of limited-vocabulary word-level translation with the open-vocabulary knowledge of sub-word units gained at the character level.

We train the word-level model on sentence pairs and we train both character models on unknown words as well as by sampling frequent words from the sentence pairs at a tunable rate. These components are learned jointly end-to-end with the additive loss function $J = J_w + \alpha J_c$ where J_w is the loss at the word-level, J_c is the loss at the character level, and α controls their relative importance. This approach removes the need for a separate UNK replacement step as in many current NMT models. We also implement an in-graph beam search decoder to keep b potential partial sentences during inference.

3.1 Word-level translation

The word-level model uses an encoder-decoder framework [8, 9] that computes an encoded representation for each source sentence. From the encoding, the decoder generates a translation by predicting one word at a time, which decomposes the sentence probability (see Equation 1).

For our parallel sentence pair corpus $\mathcal{D}$, we train our model by minimizing the cross-entropy loss:

$$J_w = \sum_{(x,y)\in\mathcal{D}} -\log p(y \mid x) \tag{2}$$

Our model uses a bidirectional encoder [10], two-layer architecture with gated recurrent units (GRU) [11], and the global bilinear attention mechanism first proposed in Luong et al. 2015 [12]. We do not use the sequence-to-sequence model provided by TensorFlow, as it lacks the fine-grained control needed for the hybrid architecture.

Rather than using raw versions of the encoder state $\dot{h}$ and the current decoder hidden state h_t to generate the next token in a sequence, global bilinear attention instead uses an "attentional state" $\tilde{h}_t$. We compute an alignment vector a_t between the encoder top-level states $\dot{h}$ and the current hidden state h_t at each sequential prediction. This alignment vector is then used to weight the source representation components $\dot{h}$. A final linear layer combines the weighted encoder representation and the decoder hidden state to produce the attentional hidden state $\tilde{h}_t$, and a softmax on the resulting activations produce our probabilities $p(y_j|y_{<j}, h_t, \dot{h})$. We include two parallel attention mechanisms so that the model has separate representations available for predicting UNK at the word-level and for generating the contextual target character sequence.

3.2 Source language character representation

To avoid discarding information when the model encounter words not in the core vocabulary, we use a multi-layer character-level encoder whose last hidden state becomes an "on-the-fly" embedding to replace the embedding for UNK. During training, we feed this model with a sub-sample of the core vocabulary words as well as the unknown words, to encourage it to learn meaningful alphabet embeddings. For computational reasons, the computation is per-type: with any set of weights, the same sequence of characters always receives the same embedding.

3.3 Target language character generation

Generation of UNK tokens is typically handled in a post-processing step by either a dictionary look-up or an identity copy. This approach suffers from alphabet mismatches between the source and target vocabularies and difficulties handling multi-word alignments. In order to address these issues, our model includes a separate recurrent model that decodes at the character-level given the current word-level state whenever the word-level model predicts an UNK. Unlike the source character-level model, the target character-level model is context-dependent and requires the current context from the word-level model to produce meaningful translations. We initialize the hidden state of this model with a separate-path attentional state from the word-level model. We train this model with a sub-sample of the core vocabulary words as well as unknown words.

3.4 Beam search decoder

During inference, we search through all the possible translations with an in-graph beam search decoder. Beam search is guaranteed to find a solution that is at least as good as the greedy solution; it does this by tracking the k best hypotheses at each timestep.

To illustrate beam search, Figure 2 shows a sample sentence "The cute cat" decoded by the beam search decoder with beam size $b = 2$. At each time step, we feed all the current hypotheses to the decoder and extend each hypothesis by the two most probable symbols, giving us four new hypotheses. Then we select the two most probable hypotheses to feed to the next timestep.

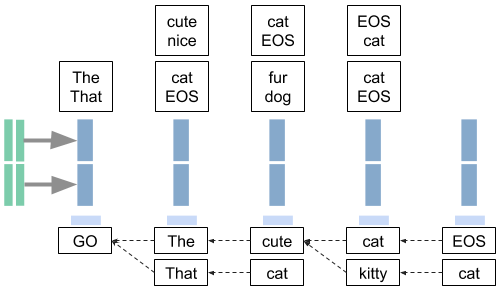

Figure 2: Beam Decoder with Beam Size 2.

4 Experiments

We evaluate our approach on Modern Standard Arabic to English translation. We compare multiple word-only and hybrid comparison models to the BLEU scores reported in Almahairi et al. [13], who provide the only known Arabic neural translation results, and who use an implementation provided by Cho that seems to be based on Cho et al. 2014 [9]. Our findings suggest substantial promise in a hybrid model.

4.1 Data

We evaluate on Modern Standard Arabic news articles. Arabic is the fifth largest language by number of speakers [14], and it is disproportionally under-studied. Arabic exhibits a complex but consistent morphological system. It has a variety of clitics, affixes, spelling ambiguities, and the root-and-pattern morphology of Semitic languages.

In order to facilitate comparison to Almahairi et al. [13] – the only Arabic benchmark we are aware of in the literature – we concatenate and train on LDC2004T18 [1], LDC2004T17, and LDC2007T08. These comprise about 1.2 million sentence pairs from news articles dated October 1998 through September 2004. For development, we use NIST OpenMT 2004, collected January 2004 through March 2004. For test, we use NIST OpenMT 2005, collected December 2004 through January 2005. No other large-scale Modern Standard Arabic parallel corpora have been collected.

We remove segments that have exact duplicates in dev and test, and we exclude from our data any segments that contain more than 50 words or any words longer than 30 characters in either language. We keep approximately 90% of the data with this preprocessing filter. Our final dataset consists of 1,087,343 training pairs, 915 dev pairs, and 946 test pairs.

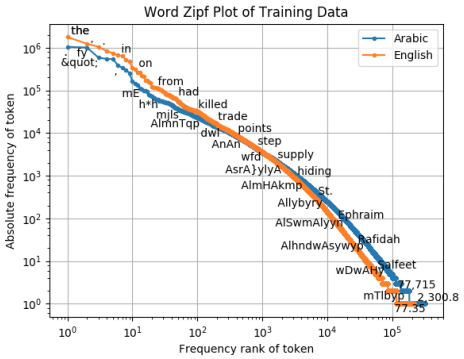

Figure 3: Zipf log-log plot of word frequencies.

As preprocessing, we transliterate from Arabic script to Latin script using Buckwalter notation [15], a common technique in Arabic NLP that makes the results more accessible to people who do not read Arabic script and ensures tools will work without additional configuration. We tokenize both languages using `tokenizer.perl` in Moses with the default settings for English, since Arabic tokens are separated by whitespace except in instances of quoting English-style abbreviations. We also perform

[1] This dataset was derived from automatically parallelizing Arabic Gigaword and reviewing with humans, and it contained different numbers of segments between the source language and reference translation, with alignment data to facilitating matching segments. We concatenated these with a space character.

orthographic normalization as per Almahairi et al. [13]; for instance, we transform all ى characters to ي.

The Arabic text contains a wider variety of tokens than English (322k unique tokens compared to 216 unique tokens in English), and its sentences are shorter (27.0m total tokens compared to 30.3m total tokens in English). Arabic has more probability in its infrequent words than English (see Figure 3), and the words in the long tail tend to be morphologically complex, alternately beginning with affixes like "Al" (*the*) or "w" (*and*) or being inflected forms like "mTlbyp" (from "Tlb", *to ask*).

4.2 Training

We use 40,000-word vocabularies for both Arabic and English, and we transform any characters that appear fewer than five times to `UNK`, leaving an Arabic alphabet of 65 characters and an English alphabet of 84 characters. We also normalize all digits to 0s. We use Adam optimization [16] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, a vertical dropout of 0.5, and gradient clipping beyond an absolute value of 5, and we weight the word- and character-level losses equally.

We train on a GPU in batches of 40 randomly selected training pairs until the training loss begins to increase, annealing an initial learning rate of 0.001 by halves by hand when the rate of loss improvement decreases.[2] We actively selected the initial learning rate as the largest learning rate that led to good activations and large but possible updates. We were able to saturate multiple models within 24 to 48 hours each with this process.

Although we began with a large model following Luong and Manning 2016 [1] (716m parameters, 240m trainable: word embeddings and hidden size of 1024, 4 layers for word model, LSTM cells), this model size proved to require 6 days per epoch – similar to the three weeks reported by Luong and Manning using their MATLAB code. As a result, we reduced the number of parameters by 80% through dropping to two layers for the word model, switching to GRU cells, and decreasing the word embedding and hidden sizes to 300 units. We initially attempted to use 16-bit precision floats to keep the same model complexity but halve its size (as per [17, 18]); however, support for 16-bit floats is not universal in TensorFlow and we had to abandon this approach. When that model trained successfully, we made the model 2.6x larger by increasing the hidden size to 768 and trained another complete model. We were able to exhaust this model's capacity as well. We expect that given additional resources, larger models would produce greater performance.

4.3 Results

We prefer the hybrid model for the theoretical reasons outlined earlier, and our results support being positive about this approach: Our model's BLEU score of 42.10 (see Table 1) achieved with saturation in less than one epoch approaches Almahairi et al.'s reported BLEU score of 48.53. The size and length of training used by Almahairi et al. are not stated in their paper, but we expect that both exceeded the series of models that we trained for this project, and thus that there is strong potential for exceeding state-of-the-art with this architecture. We have further evidence that larger models with longer training times can produce even more stellar results in our earliest word-only baseline model, which we trained with 4 layers and embeddings and hidden sizes of 1024 units, and which achieved a BLEU score of 47.07 ± 2.13 in only 3.5 epochs.

5 Analysis

We perform qualitative analysis of the model, including exploring the types and frequencies of its errors and successes. A human analysis indicates that around three quarters of the translations convey the main idea from the source sentence, though in about one in five sentences, the model hallucinates entirely new information. We also find in this analysis that the attention mechanism is important in facilitating the understanding of the sentences, and we are able to verify that the source language character model is building meaningful representations from characters in the same embedding space as the word-level model.

[2] Although the word-level model could have trained with slightly larger batch sizes, the `UNK` and frequent word sampling led to some stochasticity in model size that we did not find a straightforward way to control in TensorFlow during training.

Table 1: Results. We report the standard deviation in each direction from dividing the predictions into ten even subgroups and calculating BLEU on those subgroups ($\pm$ 1 standard deviation). The stopping criterion for all these models was saturation.

Model	Word Layers	Embedding Size	Hidden Size	Attention	Epochs	BLEU
Hybrid: Small	2	300	300	Luong	1.7	40.21 ± 1.01
Hybrid: Larger	2	300	768	Luong	0.7	$\mathbf{42.05 \pm 1.13}$
Word-only	2	300	768	Luong	1.4	42.02 ± 1.57

Table 2: Example translations illustrating understanding of word-order, extended inference length, additional date-time sequence hallucination, and common Arabic-English grammatical mistake with "mn*" (*since*). Out-of-vocabulary words for the model appear in angle brackets.

source	wqAl strw AyDA An bryTAnyA stsAEd lybyA Ely AzAlp brAmjhA ltTwyr AslHp dmAr $Aml kmA ⟨ stsAEdhA ⟩ Ely AlEwdp Aly AlwkAlp Aldwlyp ll-TAqp Al*ryp .
human	Straw also said that Britain would help Libya in ending its programs to develop weapons of mass destruction and would help its return to the International Atomic Energy Agency .
model	Straw also said Britain would assist Libya to remove its programs to develop weapons of mass destruction and return to the International Atomic Atomic Energy Agency (IAEA) . (IAEA) . " There will be no doubt . " He also said that Britain would contribute to
source	Alr}ys Alswry yzwr trkyA fy Awl zyArp mn nwEhA mn* 00 EAmA
human	Syrian President Visits Turkey in First Visit of Its ⟨ Kind ⟩ in 00 years
model	Syrian Prime Minister Rafic Hariri will visit Turkey on his first visit since 00 years ago . NNNN 0000-00-00 00 0000-00-00 00 : 00 0000-00-00 00 0000-00-00 00 : 00 0000-00-00 00 : 00 0000-00-00 00 0000-00-00 00 : 00 0000-00-00 00 : 00 0000-00-00 00 0000-00-00 00 : 00

5.1 Qualitative analysis and error analysis

We conduct a blind hand-analysis of 30 inferences selected at random to gain quantitative and qualitative insights into the performance of the model. We find at the highest level that approximately 77% of the translations convey the main idea from the source sentence, but that 43% of them engage in unnecessary repetition, 37% include at least some amount of word salad, and 17% hallucinate entirely unsupported information, such as that there were no casualties in an armed clash. Sample translations appear in Table 2.

The model has a tendency to predict periods without `EOS`. We suspect there are two causes to this behavior. First, the English language data include 89,213 examples in which periods are followed by additional text (primarily, but not exclusively, the `quot;` token). Second, the beam search decoder realizes scores cannot be maximized when `EOS` is predicted, and it takes advantage of the potential ambiguity regarding what follows a period.

In addition to postponing prediction of `EOS`, the model has unnatural tendency to generate date-time sequences following its contentful translations. This behavior occurs in about 17% of the translations. This is the result of a particular target training data source, which regularly ends in sequences like "NNNN 2003-02-10 08 : 36 : 45 2003-02-10 08 : 36 : 46 1598". Although there are only 29 instances of this kind of sequence in over one million examples, the model learns it as a unit and applies it when it expects the initially generated sequence is similar to the shorter, headline-like segments that it observed with this sequence during training.

The model also notices and ignores the idiomatic use of the prefix "w" (*and*) to begin verb-fronted sentences; this use of *and* should rarely be translated directly. The model is also clever enough to realize that when the less-common word "AyDA" (*also*) is part of the source sentence, the sense of addition should be included in the translation. It seems to have no trouble with Arabic's use

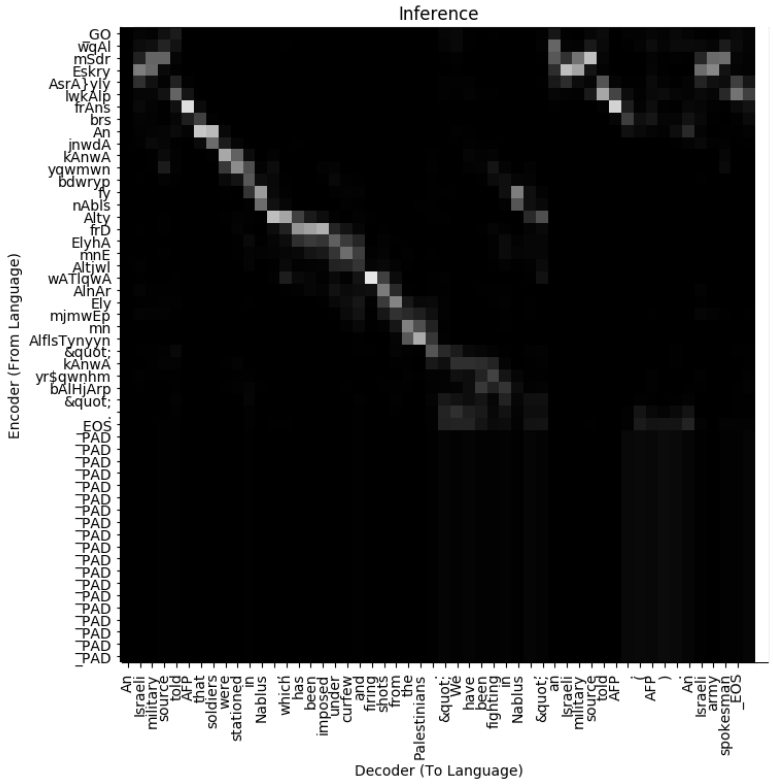

Figure 4: Sample attention alignments. The reference translation is "An Israeli military source told Agence France Presse that soldiers were on a patrol in Nablus, where a curfew has been imposed, and opened fire at a group of Palestinians 'throwing stones at them'."

of verb-subject-object word order or with adjective phrases following nouns, although English is grammatically different on both counts.

5.1.1 Attention

The attention mechanism is central to translation success. Figure 4 illustrates that attention reflects the reversed word order of Arabic compared to English at the beginning of the sentence ("wqAl mSdr Eskry AsrA}yly": *and-he-said source military Israeli*). The model pays close attention to key words like "frAns" (*France*), "wATlqwA AlnAr" (*release fire*), and "AlflsTynyyn" (*the-Palestinians*).

Grammatically, we also notice the model performing correct one-to-many transitions from Arabic to English, such as the relative pronoun "Alty" in "nAbls Alty frD" (*Nablus that imposed*) being matched to both a comma and the relative pronoun "which" in English, and "AlflsTynyyn" (*the-Palestinians*) being matched to both "the" and "Palestinians". We have further evidence of the importance of the attentional alignments when we note that "bAlHjArp" (*with stones*) receives minimal attention during inference and does not appear in the inferred translation.

5.1.2 Embedding quality

To assess whether the source language model is adequately building embeddings for unknown words, we qualitatively evaluate it by visualizing the character-derived and learned word unit embeddings of a sampling of words with t-SNE [19] (see Figure 5). From the visualization we see that the method performs reasonably well with more complicated words: it brings together nouns like "Alr}ys" (*the-president*), preposition-noun pairs like "bAglAq" (*with-shutdown*), and definite adjectives like "AlmtwqE" (*the-expected*).

The model fails to bring together shorter function words, however, such as "kmA" (*as-such*), "Alty" (*that*), "mn" (*from*), "h*A" (*this*), "fy" (*in*), and "Aly" (*to*). In some sense, the failure to adequately capture shorter words is unsurprising: with two-to-four-length sequences as input, the resulting activations cannot be as precise.

However, we suspect these results illustrate an underlying flaw in the sampling method for words with known embeddings; sampling words with known embeddings at a rate of k from each sentence will over-represent high-frequency function words, which occur in many sentences. Highly frequent words tend to be morphologically unmarked and minimally informative as to the structure of the entirely unknown words for which we are bootstrapping representations. We experimented with and recommend training the source character model using the k most infrequent in-vocabulary words in each sentence. This approach shifts the distribution of sampled words to match our expectations about unknown words as closely as possible to speed convergence, and it does not add substantial computational burden.

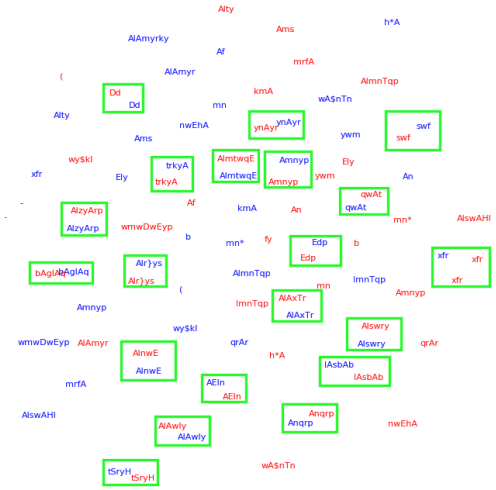

Figure 5: Similarities between unit-level word embeddings and character-derived embeddings using t-SNE. The word-level embeddings are colored blue, and the character-derived embeddings are colored red.

6 Conclusion

We implement a neural translation model in TensorFlow that is theoretically appealing in its freedom from dependency on the assumption that words and morphemes are equivalent, and which performs just as well with less training time than a word-only model. We achieve approximately state-of-the-art results for Arabic-to-English translation, and we suspect that with a larger model, our architecture would achieve better results than have previously been reported. We would like to extend this model to English-to-Arabic translation; in particular, we suspect that introducing a character-level attention mechanism would facilitate generating Arabic's root-and-pattern plus affix morphology.

Acknowledgments

We would like to thank Ignacio Cases in particular and the CS224N staff in general for valuable feedback and guidance through the preparation of this work, and Microsoft Azure for donation of resources without which this project would not have been possible.

References

[1] Minh-Thang Luong and Christopher D Manning. Achieving open vocabulary neural machine translation with hybrid word-character models. *arXiv:1604.00788*, 2016.

[2] Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W. Black. Character-based neural machine translation. *CoRR*, abs/1511.04586, 2015.

[3] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. Character-aware neural language models. *CoRR*, abs/1508.06615, 2015.

[4] Jason Lee, Kyunghyun Cho, and Thomas Hofmann. Fully character-level neural machine translation without explicit segmentation. *CoRR*, abs/1610.03017, 2016.

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

[6] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909, 2015.

[7] Thang Luong, Ilya Sutskever, Quoc V. Le, Oriol Vinyals, and Wojciech Zaremba. Addressing the rare word problem in neural machine translation. *CoRR*, abs/1410.8206, 2014.

[8] Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. Seattle, October 2013. Association for Computational Linguistics.

[9] Kyunghyun Cho, Bart van Merriënboer, Çağlar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua 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, Doha, Qatar, October 2014. Association for Computational Linguistics.

[10] Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45, November 1997.

[11] Junyoung Chung, Çağlar Gülçehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv e-prints*, abs/1412.3555, 2014. Presented at the Deep Learning workshop at NIPS2014.

[12] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. *CoRR*, abs/1508.04025, 2015.

[13] Amjad Almahairi, Kyunghyun Cho, Nizar Habash, and Aaron Courville. First result on Arabic neural machine translation. *arXiv:1606.02680*, 2016.

[14] Gary F. Simons and Charles D. Fennig, editors. *Ethnologue: Languages of the World*. SIL International, 20 edition, 2017.

[15] Nizar Habash, Abdelhadi Soudi, and Timothy Buckwalter. On Arabic transliteration. In *Arabic computational morphology*, pages 15–22. Springer, 2007.

[16] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[17] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Low precision arithmetic for deep learning. *CoRR*, abs/1412.7024, 2014.

[18] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *CoRR*, abs/1609.07061, 2016.

[19] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.