VaLaR NMT: Vastly Lacking Resources Neural Machine Translation

Minhyung Kang
Stanford University
dankang@stanford.edu

Kais Kudrolli
Stanford University
kudrolli@stanford.edu

Abstract

Neural Machine Translation (NMT) has shown great success recently, outperforming Statistical Machine Translation in most tasks. Yet, it is widely recognized that for a reasonable performance of NMT massive amounts of parallel data are required. We focus on extremely low-resource setting, where we are limited to less than 10k parallel data and no mono-lingual corpora. We look at an Elvish language Quenya created by the author J. R. R Tolkien. First, we create a character-decoder-based seq2seq NMT model as a baseline and compare its performance on various levels of data scarcity. Then, we explore the performance benefit of transfer learning by training a model on a different language. We also perform an ablation study to investigate the effect on transfer learning of freezing different layers. Lastly, we use language models and a noisy dictionary to augment our training data. Utilizing both transfer learning and data augmentation, we see a 1.5 BLEU score improvement over the baseline.

1 Introduction

With the rise of neural networks, we are seeing great improvements across different natural language processing tasks, including summarization, question answering, sentiment analysis, and translation. For example, we have seen remarkable performance boosts in translation quality since Google adopted an NMT model which learns from millions of examples [3]. It is widely recognized that for any machine learning algorithm, especially neural network-based approaches, we see better performance with more data [2]. Still, there are cases when we simply lack data due to high procurement cost or lack of availability. Hence, there has been great interest in tackling this issue of data scarcity across different machine learning tasks, including machine translation. However, the level of scarcity differs across authors: some consider 0.1 million data points scarce while others work with a few thousand parallel examples but with millions of monolingual examples.

In this work, we seek to address a level of scarcity we describe as 'vastly lacking resources'. We choose Quenya, an Elvish Language created by the author J.R.R. Tolkien, and attempt to translate it to English. While fictional, the language was carefully devised by the author, who cared greatly about the aesthetics and euphony of its language and grammar. Quenya was constructed as a "Elvenlatin", phonologically based on Latin with influences from Finnish, Welsh, English, and Greek [8]. However, since the language is fictional, the amount of Quenya data we can obtain is limited. We work with less than $10,000$ parallel examples, no monolingual text, and a few thousand-word dictionary.

We first investigate the performance of models across different scarcity levels of data. Then, we perform transfer learning using a model trained on a Finnish-English data set. We study the effects of different transfer learning settings on the quality of translation. Finally, we use language models trained on a large English corpus and a small, noisy Elvish-to-English dictionary to generate new examples to augment our data.

Preprint. Work in progress.

2 Related Work

One of the most prominent methods to deal with data scarcity in translation is transfer learning, where one initially uses a "parent" (high-resource) language to train an NMT model, then freezes certain parameters and tunes the model with a "child" (low-resource) language data set [9]. The intuition behind this is that the pre-trained model will have learned useful features from the parent language that are applicable to the child language and only needs to be tuned by the child's data set. This fine-tuning can also be seen as a regularization of the pre-trained model so that it can generalize to the child language. Similarly, sharing information across different languages for which there is abundant data has proven to be effective. Sharing lexical and sentence-level representations across multiple source languages into one target language [4] or sharing parameters across different language pairs in Google NMT [3] are some successful approaches. Some propose learning to translate by training the model from mono-lingual corpora [5, 6]. As noted previously, our setting is different in that we assume we lack a mono-lingual corpus. With the recent increased interest in reinforcement learning, some works pose the low-resource language translation as a meta-learning problem, where the language pairs are the tasks and the model learns to solve low-resource tasks with experience from high-resource tasks [10]. There is also a more data-driven rather than model-driven approach: using language models [11] or back-translation from mono-lingual corpora [1], new sentences can be generated to augment the small training data with a new, albeit noisy, parallel corpus. In this work, we utilize the transfer learning scheme and explore the idea of data augmentation.

3 Approach

3.1 Neural Machine Translation

Our main NMT model is a seq2seq network. The encoder portion is a bidirectional LSTM that takes in embedding vectors of the source sentence (either in Quenya or Finnish, in our case), and the decoder is a unidirectional LSTM that is trained on embedding vectors from the labeled target sentence. All embeddings in the model are learned from random initialization. The model uses multiplicative attention so that at each time step the decoder can choose what context of the encoded source sentence to "attend" to. The decoder output at each time step is concatenated with the attention output and passed through a softmax function to produce a probability distribution for the next word in the target sentence. Finally, the model uses the cross-entropy loss function.

3.2 Character-Based Neural Machine Translation

The base NMT system above is improved with character-based encoding and decoding. The character-based encoder [15] first looks up an embedding for each character and then combines them into a word embedding using a CNN followed by a highway network. The character decoder [16] is used when the word-level decoder produces an out-of-vocabulary word. The decoder output at that time step is fed into a unidirectional LSTM that uses greedy decoding to predict characters at each time step. These character-based components are particularly helpful for low-resource NMT as a small training data set increases the chance the model encounters out-of-vocabulary words. With character encodings it can still encode an unseen source word and decode an unseen target word.

3.3 Transfer Learning in a Low-Resource Setting

Low Resource We are particularly interested in improving the seq2seq and character-encoding model described above for low-resource machine translation, that is when we do not have many parallel examples to train the model on a source language. With very few parallel examples (approximately $8,000$) and no larger corpus for Quenya-English, it is difficult for our unaltered NMT model to generalize to unseen text from that language.

Transfer Learning Transfer learning has proved to be effective in various fields [14]. In its most basic form, we first perform training on a set of tasks $\mathcal{T}$ to learn model parameters that minimize the loss function $\hat{\theta} = argmax_{\theta}\mathcal{L}(X_{\mathcal{T}}, Y_{\mathcal{T}}; \theta)$. Then, we use $\hat{\theta}$ as the initialization of parameters to train on another task, $\mathcal{T}'$. We can adjust training processes as needed (e.g. freezing certain layers' weights). This framework is beneficial in low-resource cases if $\mathcal{T}$ and $\mathcal{T}'$ share some context and has been tried several times in low-resource NMT settings [9].

Table 1: Data used for experiments

Data Set	Source	Target	Training	Validation	Testing
Bible	Elvish	English	7,135	396	396
Misc	Elvish	English	N/A	N/A	215
Europarl v8	Finnish	English	1M	1,000	3,000
Newstest 2018	Finnish	English	N/A	N/A	3,000

4 Experiments

4.1 Data

All the experiments were carried out on the data sets described in Table 1 and their subsets. We use the Finnish-English data set because Finnish was one of the languages that inspired the development of Quenya [8], and it is a larger data set that can be constrained to our desired scarcity and used to compare our model with available baselines by other authors. We note that $(i), (ii)$ and (iv) are data sets found online and have not been necessarily verified by a reliable community.

(i) Bible: Our main source of Elvish data is a Quenya translation of the whole New Testament[17]. The Bible has been a consistent source of data in NLP due to its availability in many languages. On the other hand, it is important to note that this translation is the work of a single person, and the integrity of work cannot be verified - there are human errors including missing lines and spelling mistakes.

(ii) Misc: We utilize other Quenya passages found in Tolkein's books [18]. While these do not provide a evaluation metric that can be compared with other literature, this data set is mostly used for analysis of our model's generalizability by looking at performance on non-biblical text.

(iii) Europarl V8: We use 1 million sentences from the Europarl corpus [12], which are texts from the proceedings of the European Parliament. We randomly select one million sentences for training and one thousand sentences for validation.

(iv) Newstest 2018: We use the official test set for WMT 2018 [13] to evaluate the performance of our Finnish-English NMT model. This comes from a well-established machine translation conference.

(v) Elvish-English dictionary: We found, downloaded, and parsed a Quenya-English dictionary [19] for use in our data augmentation. This dictionary has about 5000 noisy Elvish-English pairings, including 2842 and 3466 unique Elvish and English words, respectively.

Data Cleaning (i) was in word documents, with each paragraph of Elvish followed by the corresponding English translation. We used a Python script to parse the document into a parallel corpus by breaking it up into verses, manually correcting errors when we encounter them, and dividing the sentences into training and test sets. Due to its small size, (ii) was manually compiled into a parallel corpus. (iii) and (iv) were both downloaded from WMT 2018 website. A Python script was used to split (iii) and take out sentences which had characters outside of our predefined set of acceptable characters. For (iv), a Python script using BeautifulSoup4 was used to parse the SGML file into a flat text file. For (v), a Python script with BeautifulSoup4 was used to parse the htm file into a dictionary.

4.2 Experimental Setup

For our experiments, we use our char-decoder model from Assignment 5 of the Winter 2019 CS224N course at Stanford University. The model is trained with an initial learning rate, which is decayed with a decay rate after our perplexity on validation set does not improve for 5 successive validation iterations, and we stop training after 5 such decays. The underlying network parameters were uniform across the experiments with an embedding size of 256 and hidden size of 256. When we did not specify initialization parameters, they were uniformly initialized as values between $[-0.1, 0.1]$. For vocabulary construction, we limit the source vocabulary words to $8,500$ to match that of our Elvish data, and limit the target vocabulary to $50,000$. A total of 127 characters (alphanumeric, punctuation, some accented characters) was selected as our universal character set, and any data which contained

characters apart from these was discarded. Most experiments were carried out using Microsoft Azure NV6 Virtual Machine with 6 vCPUs, 56GB RAM, and 1 MC60 GPU.

Table 2: Baseline performance on Elvish

Corpus size	BLEU	
	Bible	Misc
7135	44.216	6.659

Table 3: Performance on Finnish by corpus size

Corpus size	BLEU	
	Europarl	Newstest
100	0	0
1000	1.274	0
8000	6.769	1.082
100,000	16.723	3.587

For evaluation, we calculate 4-gram BLEU scores on in-context and out-of-context data sets. In-context texts are test data separated from the corpus on which the model was trained while out-of-context only shares the same language and is from a different data set. This allows us to evaluate how well the model captured the training data as well as its generalizability.

4.3 Performance of Models under Low-Resource Constraints

Initial evaluation of model performance for Elvish corpus

To establish baseline performance of our character-encoder model, we train the model on just the Elvish *Bible* data set for 30 epochs with learning rate 0.001, decay rate of 0.5, batch size 32, and a dropout rate of 0.3. As shown in Table 2, the model does very well on the test examples taken out of the Bible. This is because the Bible is repetitive, often using the same phrases and vocabulary (See Appendix A, Ex 1). However, on the test set that is a collection of Quenya from other sources, the model does not perform well, showing that there is not enough data in the Bible training set to generalize to different text sources. One observation that makes this particularly clear is with names. The model translates unseen names as Biblical names (See Appendix A, Ex 2).

Comparison of model performance for low-resource languages

To further test our model with low-resource data sets of different levels, we train it on our Finnish *Europarl* data set and simulate a low-resource language by constraining the data set to a low number of examples as done in [9]. Then, we rebuild the vocabulary based on the constrained data set. We try this at four levels of scarcity: 100, 1000, 8000, and 100,000 (we choose 8000 as opposed to $10,000$ because it is closer to the number of examples in our Elvish data set). We still validate the training on our validation set of 1000 examples from *Europarl*, and we also test the data on our second hold out set from *Europarl* and test set from *Newstest* (each of these are 3000 examples). Each experiment is run with learning rate of 0.001, decay rate of 0.5, dropout rate of 0.3, and batch size 16 for 30 epochs.

From Table 3, we see that 100 and 1000 examples are far too small to make coherent translations on *Newstest*. Above that scarcity level, the model is able to translate and improves on both *Europarl* and *Newstest* as the training data increases. The performance of the model on training size 8000 and on the *Europarl* test set is comparable to its performance on the Elvish data tested on the Bible hold out.

4.4 Performance of Transfer-Learning Models

The results of transfer learning are in Table 4. For the source model, we trained our char-decoder network on the Finnish *Europarl* data set. We used batch size of 16, dropout value of 0.2, and ran until it early stopped around 10 epochs. Then, we performed transfer learning using batch size 32 and dropout rate of 0.5; note the baseline model was trained using same parameters. We tried different intializations of vocabulary. We either retained the source or target vocabulary and character embeddings of the original model (denoted as O in Table 4) while letting it train the word embeddings, or we used the same initial vocabulary as the baseline model (contains the most frequent words in the text). It is important to clarify that the source vocabulary here includes character embeddings while the target vocabulary includes the target input word embeddings and output character embeddings. We tested the models on in-context text (*Bible*) and out-of-context text (*Misc*).

We make a few observations regarding performance and vocabulary initialization. First, as we are using character-level embeddings in the source, copying over word-level vocabulary does not have too much effect. However, copying over the target vocabulary has rather arbitrary effects rather than

Table 4: Performance of transfer learning models compared with baseline

Model	Source Model	Source Vocab	Target Vocab	BLEU	
				Bible	Misc
Transfer-10	Finnish-10	X	X	**61.401**	9.327
Transfer-10-Src	Finnish-10	O	X	61.129	8.327
Transfer-10-Tgt	Finnish-10	X	O	60.382	7.872
Transfer-10-Full	Finnish-10	O	O	60.655	9.128
Baseline	N/A	N/A	N/A	60.108	**9.890**

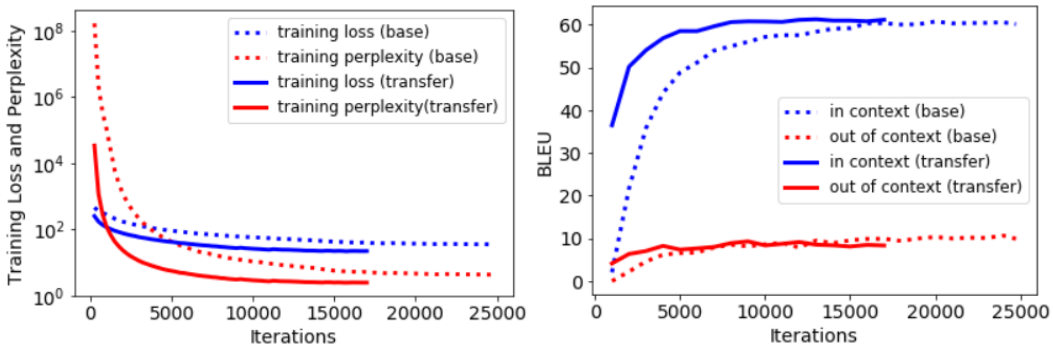

Figure 1: Training of base model and Transfer-10-Src over iterations. Training loss (left) and perplexity (right) BLEU scores for two different test sets

a direct causal relationship. Overall, we observe that transfer learning can boost BLEU scores on in-context texts while surprisingly hurting performance on out-of-context tasks.

Figure 1 shows the differences in training between baseline model and Transfer-10-Src model. As we expected, on the left the transfer learning models converges much faster than the base model. On the right, we see observe better BLEU scores initially with transfer learning, but the baseline model catches up eventually and even surpasses in out-of-context tasks.

4.5 Ablation Study

We followed the experiment carried out in [9] and performed an ablation study on the *Transfer-10-Src* model from Table 4; the result is described in Table 5. We do not experiment with freezing the attention model, encoder, and word decoder as [9] points out this hurts the performance. We observe that freezing target input embeddings and char embeddings give us the best training loss and BLEU score on in-context data set. This may be because the model utilizes the structure of language from before yet allows modification to fit to our training set. On the other hand, we achieve the highest out-of-context BLEU score if we freeze the whole char decoder while getting a relatively high training loss and low performance on the in-context data set. This is expected as we are carrying over and retaining information from a bigger data set (*Europarl*). It is worth noting that freezing source char embedding does not hurt our performance too much, for Finnish and Elvish shared the same set of alphabets and we do not necessarily lose any information by freezing it.

4.6 Data Augmentation

As in other deep learning settings where there is limited data, we augment our data, i.e. we transform our existing examples sentence pairs into new sentence pairs to increase the size of our training set.

Candidate Generation For a given English training sentence, we use the method from [11], which generates replacement candidates for each word in the sentence using two LSTM language models (LMs), a forward and a backward. The forward LM is passed the normal sentence, and the backward model is passed the reversed sentence so it learns to predict the sentence backward. The idea is that at each time step the hidden state of the LSTM encodes the likelihood of each word in the vocabulary appearing in the context of the rest of the sentence instead of the actual word at that time step.

Table 5: Ablation study of transfer learning

Source Char Embedding	Target Input Embedding	Char Decoder	Target Char Embedding	Loss Train	PPL Train	BLEU Bible	BLEU Misc
🔒	🔒	🔓	🔒	24.99	2.80	61.34	8.94
🔒	🔓	🔓	🔒	23.30	2.60	61.47	10.20
🔒	🔓	🔓	🔓	23.49	2.62	61.34	9.52
🔓	🔒	🔒	🔒	50.92	8.13	60.88	9.48
🔓	🔒	🔓	🔒	**22.15**	**2.48**	**61.78**	9.11
🔓	🔓	🔒	🔒	44.18	6.14	61.35	**10.41**
🔓	🔓	🔓	🔒	22.49	2.52	61.39	9.58
🔓	🔓	🔓	🔓	22.34	2.50	61.13	8.33
Baseline				35.71	4.33	60.11	9.89

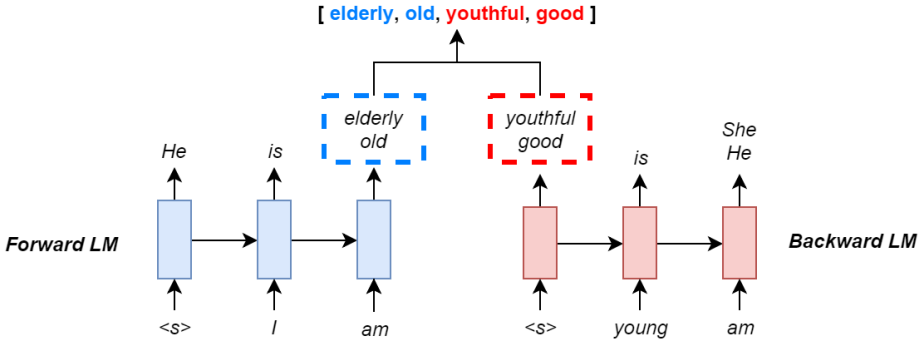

Figure 2: How the forward and backward language models are used to generate candidates to replace the word 'young' in the sentence 'I am young'. The outputs of the LMs are combined into one candidate list, shown at the top.

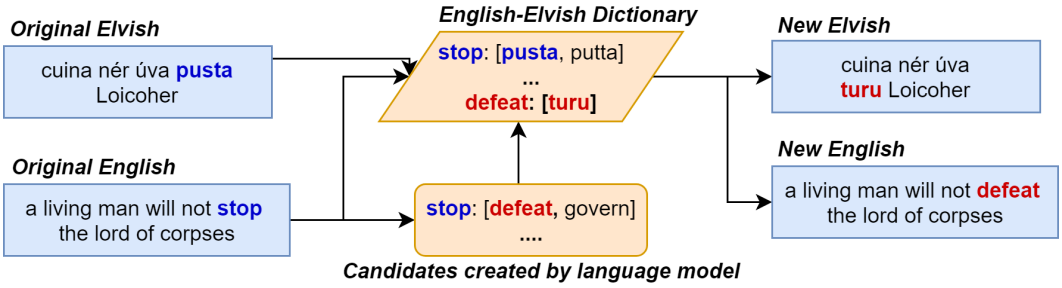

Figure 3: Creating new examples with candidates created from language model and English-Elvish dictionary. We replace 'stop' with 'defeat' and make the corresponding replacement in Elvish.

Each of our language models is a unidirectional LSTM of hidden size 256 with word embeddings also of size 256. We drop out on the word embeddings and on the outputs of the LSTM with a dropout rate of 0.5. Each LSTM is trained using a log softmax loss function all the way back to the word embeddings on our *Europarl* English training data.

As shown in Figure 2, we generate replacement candidates by running our transfer learning model's training data through the trained forward and the backward LMs. For each time step of the LM, we choose the k (where k is a hyperparameter) hidden outputs with the highest probabilities, lookup their corresponding words in the vocabulary, and set those words as the candidates. Then, we take the union of the candidates from the forward and backward LMs at each time step.

Example Generation Next, we use the candidates generated from previous step and augment our data. While we now have possible replacements of English words, we do not know 1. which Elvish

word correspond to word we want to change in English 2. Elvish translation of new English word we want to use. In [11], they use a fast-align model [7] to find alignment phrases of parallel-corpus and make the replacement. However, we take a rather brute-force method using our Elvish-English dictionary.

Table 6: Data augmentation experimentation results. 'k' is the number of candidate words each of the forward and backward language model can generate for each word in the original sentence.

Augmentation Method	**k**	**Loss Train**	**PPL Train**	**BLEU**	
				Bible	**Misc**
Dictionary	N/A	36.12	9.715	**61.37**	10.45
Ignore First 6000	100	44.84	5.808	60.31	9.150
Ignore First 6000 + Dictionary	100	40.38	8.084	60.72	8.680
Filtering	5	38.87	**4.405**	58.67	8.320
Filtering	3	46.52	6.692	61.27	9.289
Filtering	1	47.74	7.131	60.96	9.597
Less Repeat	3	47.95	7.109	61.02	8.797
Less Repeat + Dictionary	3	**35.62**	8.448	61.10	10.69
Less Repeat + Replace Original	3	48.70	7.516	60.83	9.708
Less Repeat + Replace Original + Dictionary	3	39.65	11.68	61.08	**11.38**
Baseline		35.71	4.33	60.11	9.89

The process of example generation is described in Figure 3. Given an English sentence, Elvish sentence, and candidate word substitutions, we make sure we have an entry in our dictionary for the word in original sentence we want to replace ('*stop*'), as well as check we have its corresponding Elvish word ('*pusta*') in the original sentence. Hence, we identify the alignment of the words to replace. Then, we iterate through our candidate word substitutions and select replacement English words we have in our dictionary ('*defeat*'). Finally, we can replace the words in English and Elvish sentences, respectively.

Experiments and Results All of our data augmentation experiments (summarized in Table 6) involved tuning the pre-trained Finnish model using our best transfer learning method.

Dictionary: As an initial basic augmentation experiment, we simply added the English-Elvish dictionary into our training set as single-word examples. This unexpectedly was one of the most effective improvements as our top 3 BLEU scores on Misc involve the dictionary. This may have helped because there are many words in the test set that are not in the original training set, and now the model would simply know these words. Similarly, during training the model may have effectively learned a direct mapping from certain Elvish words to English words, which means there is very little probability these words would ever be mistranslated.

Ignore First 6000: Additionally, we experimented with limiting the vocabulary to a subset that could be selected as the top k to promote selection of rarer words not otherwise in our vocabulary. First, we ignored the first 6000 words in our vocabulary as these came from the New Testament portion of our data. This hurt our performance even with the addition of the dictionary. It's likely that using k=100 in these runs created too many repetitive examples that caused the model to overfit to certain sentence structures and dilute the effects of the improvements. From these results, we concluded lower k's are better.

Filtering: We also tried filtering the vocabulary, such that we only selected words that were in our English-Elvish dictionary and in our vocabulary but not in our New Testament data. We found this method allows many very common words to be selected over and over again. To limit the number of times a word is used we kept track of each word's usage and scaled their probabilities down by this word count. This still caused performance degradation as there were still repeated sentences with one or a few words changed.

Less Repeat: Filtering still resulted in repetition because if a word had multiple candidates for replacement we would create multiple duplicated examples with a single word replaced. We felt this might be diluting the effectiveness of introducing new words in new contexts. Instead, if we had multiple candidates for a given word, we would take the word between forward and backward with the highest probability and just generate one new example with it. In all of our "Filtering" and

"Less Repeat" runs, we replace all the words at once and create one additional training example out of replacements for each word. Again, striking the balance between adding augmented examples and not repeating examples proved difficult because without the dictionary having fewer examples to reduce repetition degraded performance.

Replace Original: Finally, we attempted to replace the example sentences from our training set with the augmented sentences instead of appending them to training set and using both. This proved effective because even though it didn't increase the quantity of training examples, it improved the quality by putting rarer words into the data, thereby helping the model generalize better. Combined with "Less Repeat" and adding the dictionary, this produced our best Misc BLEU score.

5 Analysis

Translation Quality We see a varying quality of translations with our best model. Example 1 in Appendix B is a fairly good translation that is almost word for word. "Possess" is perhaps not the word an English speaker would use even though it is a synonym of "have" but is fairly close. The model may produce a rarer word than "have" because of our rare word augmentation. Example 2 first shows that the model struggles with named entities. It also shows that since there is not enough New Testament training data, some of the translation still reflect the Finnish-English Europarl training. "German" and "state" are likely words that were learned during the source model training and not unlearned during target model. Example 3 shows that in some cases the model produces translations that hardly correlate to the golden translation. We see that the model's translation does somewhat posses the list structure of the sentence but doesn't translate any of the words in the list properly.

Data Augmentation We found that our data augmentation created the most fluent English sentences when it replaced nouns with nouns. From Example 1 in Appendix C, we see it replaces "peace" with the equally appropriate "prosperity". It does an average job with articles, and verbs in Examples 2 and 3, respectively. It replaces the article "the" in Example 2 with "southern", which makes sense in the context of city but is not fluent. In Example 3, it replaces a verb with another verb, but here it does not quite makes sense to "react to the ground". In Example 4, the augmentation does poorly with prepositions, replacing "if" with "core", likely because we are always trying to substitute rare words and are unlikely to get prepositions as candidates. One limitation is that as we are not fluent in Elvish, we were not able to validate Elvish word replacement. We assumed that we could replace Elvish words with their dictionary form; however, due to noun declensions and verb conjugations this may not have always been exactly correct. We reasoned that because our encoder works at character level, even an improperly inflected word form would be close enough to be translated.

Evaluation Another crucial component of working with low-resource setting is that we also lack testing data. That means it is tough to evaluate the performance of our models. As its critical to have as much data as possible, we couldn't afford to use many data points for testing. Given such a small test set (200 sentences), there is an uncertainty with regards to performance of model; is the model actually performing better/worse, or did it get lucky with the 200 sentences? If an idea is applicable to another language, we can carry out the experiments in a synthetic low-resource data and test the result with an abundant test set.

6 Conclusion

To improve the model in the future we could combine the forward and backward language model in different ways to generate candidates, such as choosing the one with the greatest sum/product of logits. We could also learn and perform Named Entity Recognition (NER) to handle unknown names more smoothly (e.g. just pass them to the translation). Lastly, we could utilize better parsing, such as normalizing cases and stemming.

In this paper, we explored different methods to build an effective NMT model with very limited data. We trained a model between Finnish and English and performed transfer learning to Elvish-English. After experimenting with freezing certain parts of layers, we were able to increase the BLEU scores compared to baseline models. Then, we utilized an online dictionary of Elvish-English and LSTM language models to generate new examples and added to our training data. We showed that transfer learning with data augmentation can improve our model performance for low-resource cases.

References

[1] Rico Sennrich, Barry Haddow, Alexandra Birch (2016) Improving Neural Machine Translation Models with Monolingual Data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*

[2] Chen Sun, Abhinav Shrivastava, Saurabh Singh, Abhinav Gupta (2017) Revisiting Unreasonable Effectiveness of Data in Deep Learning Era. In *Proceedings of the 2017 Conference of International Conference on Computer Vision*.

[3] Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, Jeffrey Dean (2016) Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. In *Transactions of the Association for Computational Linguistics - Volume 5, Issue 1*.

[4] Jiatao Gu, Hany Hassan, Jacob Devlin, Victor O.K Li (2018) Universal Neural Machine Translation for Extremely Low Resource Languages. In *Proceedings of the 2018 Conference of North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

[5] Mikel Artetxe, Gorka Labaka, Eneko Agirre, Kyunghyun Cho (2018) Unsupervised Neural Machine Translation. In *Proceedings of the Sixth International Conference on Learning Representations*.

[6] Guillaume Lample, Alexis Conneau, Ludovic Denoyer, Marc'Aurelio Ranzato (2018) Unsupervised Machine Translation Using Monolingual Corpora Only. In *Proceedings of the Sixth International Conference on Learning Representations*.

[7] Chris Dyer, Victor Chahuneau, 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*.

[8] Carpenter, Humphrey; Tolkien, Christopher, eds. (1981) The Letters of J. R. R. Tolkien. London: George Allen & Unwin. ISBN 978-0-04-826005-5. No 144.

[9] Zoph, B, Barret Zoph, Deniz Yuret, Jonathan May, Kevin Knight (2016) Transfer Learning for Low-Resource Neural Machine Translation In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

[10] Jiatao Gu, Yong Wang, Yun Chen, Kyunghyun Cho, Victor O.K. Li (2018) Meta-Learning for Low-Resource Neural Machine Translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

[11] Marzieh Fadaee, Arianna Bisazza, Christof Monz (2017) Data Augmentation for Low-Resource Neural Machine Translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*.

[12] Philipp Koehn (2005) Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of the tenth Machine Translation Summit*.

[13] Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, and Christof Monz (2018) Findings of the 2018 Conference on Machine Translation (WMT18). In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*.

[14] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu (2018) A Survey on Deep Transfer Learning. In *Proceedings of the 27th International Conference on Artificial Neural Networks*.

[15] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush (2018) Character-Aware Neural Language Models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*.

[16] Minh-Thang Luong, Christopher D. Manning (2016) Achieving Open Vocabulary Neural Machine Translation with Hybrid Word-Character Models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

[17] The New Testament in Neo-Quenya. `https://folk.uib.no/hnohf/nqnt.htm`. Accessed February, 2019.

[18] Texts in Quenya. `https://en.wikibooks.org/wiki/Quenya/Texts_in_Quenya`. Accessed February, 2019.

[19] Quenya-to-English wordlist. `https://folk.uib.no/hnohf/quen-eng.htm`. Accessed 2019 March.

A Examples of char-level decoder on *Bible*

Example 1 - Repetition in the Bible

Train set example:
And Jesus said to them: "The friends of the bridegroom surely cannot have grief while the bridegroom is with them?..."
Test set example:
And Jesus said to them: "Surely the friends of the bridegroom cannot have a fast while the bridegroom is with them?..."

Example 2 - Replacing unknown names with Biblical ones

Label sentence:
Farewell - **Galadriel's** lament in **Lórien**
Base model output:
Blessed from **Nazareth** has prepared in **Egypt**

B Translation Examples

These examples all come from the Miscellaneous (out-of-context dataset) that is our main benchmark for model performance.

Example 1 - Good example, close translations

Elvish: Mernelye fire ar harya alcar.
Gold Translation: You wanted to die and have glory.
Our Translation: You wanted to die and possess glory.

Example 2 - OK example, base Europarl training data leaks into Elvish translation

Elvish: A Aina Faire Eru órova omesse.
Gold Translation: O Holy Spirit, God, have mercy on us.
Our Translation: O German Spirit have mercy on a state.

Example 3 - Bad example, little correlation with gold sentence Elvish: Náro pold', úmea, morn' ar alta.
Gold Translation: He is strong, evil, and dark.
Our Translation: A man's street, an abuse, 'favours and unwise.

C Data Augmentation Examples

These examples come out of our best augmentation model: reduced repetition in examples combined with the Elvish-English dictionary and replacing the original training examples with new ones.

Example 1 - Good replacement of noun

Original: For God's kingdom is not eating and drinking, but justice and **peace** and joy by Holy Spirit.
Augment: For God's kingdom is not eating and drinking, but justice and **prosperity** and joy by Holy Spirit.
Original: An Eruo aranie ua matie ar sucie, mal failie ar **raine** ar alasse Aire Feanen.
Augment: An Eruo aranie ua matie ar sucie, mal failie ar **aute** ar alasse Aire Feanen.

Example 2 - OK replacement of verb

Original: But are not two sparrows sold for a piece of copper? And yet one among them does not **fall** to the ground without your Father's knowledge.
Augment: But are not two sparrows sold for a piece of copper? And yet one among them does not **react** to the ground without your Father's knowledge.
Original: Ma uat filit atta vácine mittan urusteva? Ananta er mici tú ua **lanta** i talamenna pen Atareldo istya.
Augment: Ma uat filit atta vácine mittan urusteva? Ananta er mici tú ua **accar** i talamenna pen Atareldo istya.

Example 3 - OK replacement of article

Original: He was a man thrown into prison by reason of an uprising that had happened in **the** city, and of murder.
Augment: He was a man thrown into prison by reason of an uprising that had happened in **southern** city, and of murder.
Original: Sé náne nér hátina mir mando castanen amortiéno ya náne martienwa **i** ostosse, ar nahtiéno.
Augment: Sé náne nér hátina mir mando castanen amortiéno ya náne martienwa **hyarna** ostosse, ar nahtiéno.

Example 4 - Bad replacement of preposition

Original: We know that the Law is good, **if** one uses it in a regular manner.
Augment: We know that the Law is good, **core** one uses it in a regular manner.
Original: Istalve in i Şanye mára ná, **qui** mo *yuhta sa mi şanya lé.
Augment: Istalve in i Şanye mára ná, **ende** mo *yuhta sa mi şanya lé.