Domain Informed Neural Machine Translation: Developing Translation Services for Healthcare Enterprise

Sahil Manchanda and Galina Grunin
Optum
{sahil_manchanda, galina.grunin}@optum.com

Abstract

Neural Machine Translation (NMT) is a deep learning based approach that has achieved outstanding results lately in the translation community. The performance of NMT systems, however, is dependent on the availability of large amounts of in-domain parallel corpora. The business enterprises in domains such as legal and healthcare require specialized vocabulary but translation systems trained for a general purpose do not cater to these needs. The data in these domains is either hard to acquire or is very small in comparison to public data sets.

This is a detailed report of using an open-source library to implement a machine translation system and successfully customizing it for the needs of a particular client in the healthcare domain. This report details the chronological development of every component of this system, namely, extraction of data from in-domain healthcare documents, a pre-processing pipeline for the data, data alignment and augmentation, training and a fully automated and robust deployment pipeline. This work proposes an efficient way for the continuous deployment of newly trained deep learning models. The deployed translation models are optimized for both inference time and cost.

1 Introduction

The emergence of Neural Machine Translation (NMT) was sparked by the use of Recurrent Neural Networks (RNN) for machine translation. The RNN encoder in this approach is responsible for encoding the source language phrase into a fixed-length vector. This vector is then decoded into the target language (Cho et. al., 2014). Some approaches also use Long Short Term Memory (LSTM) (Hochreiter, 1997) for this task (Sutskever et al., 2014).

Sequence to Sequence models with attention (Bahdanau et al., 2014) started coming to fruition with the advent of the idea that the LSTM layers stacked on top of each other stopped improving at about a depth of 4. The attention-based models were essentially of two types, local and global. The attention models proved to be much more efficient in translating long sentences as compared to non-attention models (Luong et al., 2015). But long sentences still exhibited "exposure bias" which led to the emergence of attention models that attend over the input and generated outputs separately (Paulus, Xiong, Socher, 2017).

Despite these remarkable advances in NMT, one of the major problems still faced by some enterprises is domain-specific translation where the general-purpose translators do not perform well. Leveraging in-domain corpora to skew a general purpose translator successfully has been an area of interest in NMT in recent years. One of the successful techniques to do this has been mixed fine-tuning (Chu, Chenhui and Wang, Rui, 2018) which suggests to train an NMT model on out-of-domain corpora until model convergence and then resume training from step 1 on a mix of in-domain and out-of-domain data.

This paper tackles the challenge of translating in-domain correspondence letters for one such healthcare enterprise. The documents that these enterprises send out to their clients contain confidential information, legal/medical clauses that need to be translated in an enterprise-specific manner. Some parts of text like entity names, addresses, text in a language different from the language of the document do not need to be translated. In addition, the human-translated data that we leverage is present in sources such as HTML pages, or .doc/.docx forms, so intelligent parsing is required to extract parallel text.

In this paper, we illustrate the construction and deployment of a translation system from English to Spanish. First, we describe the training from the ground up of a general-purpose translator using an open-source library, OpenSeq2Seq (Ginsburg et al., 2018). Then, we describe an iterative process to customize this model for two use cases: medical handwritten text and formal medical correspondence letters. We use various resources from our clients at our disposal. First, we scanned the human-translated documents that were sent to the end-users to extract domain-specific sentences in the source and reference languages, cleaned the data, and built a parallel corpus. We also utilized another resource obtained from the enterprise, a translation memory, which is an XML to XML mapping that human translators used to refer to for the translation of specific sentences and form sections of a letter. We used these domain-specific parallel corpora to fine-tune the general-purpose model while making sure it did not overfit.

This study also proposes continuous deployment architecture for these models that is highly efficient at inference time and seamlessly deploys newly trained models with zero downtime.

The rest of the report is organized as follows:

Section 2 describes all the data sets used in this work, section 3 outlines how the data from different sources was made suitable for training, section 4 describes the architecture and evaluation procedures used. Section 5 explains the experiments and customization of the model towards the target domain, section 6 describes an efficient and inference-optimized architecture for our model and we conclude in section 7.

2 Data sets

Our data sets included two sources from the public domain, one used for training and one for evaluation for the general domain. For the customization part of the training process, various resources internal to the customer were leveraged. The customer sends medical claim correspondence letters which are manually translated in Spanish. We used the translations from these documents as a major part of our in-domain data set. Also, human translators use a translation memory to refer to the correct translation of some phrases or sentences. Translation memory consists of XML mapping files that contained source XML in English and reference XML in Spanish. Table 1 summarizes the size and source of the data sets used in this work. Note that the size of correspondence letters and translation memory is measured before they are extracted and pre-processed (3).

Data Fragment	Source	Sentences
Paracrawl [1]	Public	38M
WMT-News	Public	14K
Correspondence Letters (M&R)	Customer Internal	492K
Translation Memory (M&R)	Customer Internal	15K

Table (1) Data sets used in this work and corresponding source and number of sentences in each.

3 Data Preparation

While our baseline experiment used the raw version of the public data, we cleaned and aligned all the data sets to ensure the quality of data on which the model is trained. This step was especially critical to the customer documents, as they were word document files (Docx) containing tables and forms and not only plain text. This needed special attention as described in the following sections.

3.1 Public Data

The Paracrawl [1] v5 open corpus data set is used for the public section of this model's training data. This data contains 38,971,347 sentences of English and Spanish. The data was subjected to

[1] http://paracrawl.eu.

the following alignment and pre-processing procedures:

1. **Sentence Alignment**
   A mismatch in the corresponding text in the source and reference language can cause the translator to learn wrong short term dependencies. Hence, the sentences in the data set were aligned using "Yet Another Sentence Aligner" (Lamraoui, Langlais, 2013) which has shown to improve the quality of statistical machine translation. The sentences which were not successfully aligned were discarded.

2. **Language Check Elimination**
   Sentences not from the intended language were eliminated.

3. **Redundant Characters Elimination**
   On closer inspection of the data, we found a lot of acronyms, and language idiosyncrasies in the provided files like '. . . ' instead of a '.'. For the purpose of a medical correspondence letter translator, it was assumed that the letter would follow the correct English language syntax. Hence, the idiosyncrasies of the text were neutralized.

4. **Data Augmentation**
   Some documents contain text with incorrect casing and punctuation, for instance, the text in a bullet list, prescriptions, forms, etc. The system has to be robust enough to endure the incorrect casing and punctuation that it can encounter in a text. So, we converted 20% of the data to its lowercase or punctuation-less form. 80% of the data remained as is.

The Paracrawl data set of size 38M sentences was reduced to 24M after the pre-processing steps were done. We have used both raw and pre-processed versions of this data set for our experiments to note the effects of these steps on translation quality. The WMT-News (J. Tiedemann, 2012) data set was kept as is for evaluation purposes.

3.2 Customer Data

We requested manually translated documents from the source to reference language from our customer. Since the customer is a medical entity sending out medical claim acceptance or refusal letters in English and Spanish, we were able to obtain 22,292 pairs of claim refusal letters that they sent to their subscribers. The text was extracted from these documents and the following operations were applied to make it ready for training.

1. **Data Extraction**
   Data was extracted from the correspondence letters of the customer. This included names, addresses, medical terminologies, law terms, etc. All confidential data was deleted and the rest of the text was utilized.

2. **Record of Untranslated Text**
   The text that was the same in both the source and reference documents was recorded for further analysis. This could be due to various reasons: some text could be personal data that should not undergo translation, or it could be medical/legal terms that should stay as they are. They can be leveraged in our systems as lookup tables to aid in translation.

3. **Sentence Alignment**
   As was done with public data, we aligned the extracted text to eliminate any mismatch between source and reference language texts. YASA (Lamraoui, Langlais, 2013) was used to align these sentences.

The final count of the sentences that were extracted from the customer documents was 323,161, reduced from 492K due to pre-processing steps. This data set was divided into two parts for our experiments, part 1, comprising of 182,143 sentences, and part 2, comprising of 141,018 sentences. The division was random and based on the order in which these documents were sent to us. Part 1 was utilized to skew our model to the customer domain while part 2 served as our in-domain test data set.

4 Model Training

4.1 Model Architecture

The OpenSeq2Seq (Ginsburg et al., 2018) toolkit for experimentation with Natural Language Processing has been used in our experiments. This toolkit has access to various sequence to sequence architectures. For experiments in this paper, we have used a transformer-based model architecture with self-attention (Vaswani et al., 2018). This model is based on an encoder-decoder sequence to sequence architecture which has been found to outperform vanilla RNNs and CNNs in terms of machine translation. The Transformer starts by generating initial representations, or vector embeddings,

for each word. Then, using self-attention, it aggregates information from all of the other words, generating a new representation per word informed by the entire context.

4.2 Evaluation

We evaluate the models on a general-purpose data set and text from in-domain customer documents. This section specifies the metric and the fractions of the data sets we used for evaluation.

4.2.1 Metric

The experiments presented in this work use the metric of BLEU Score (Papineni, Roukos, Ward and Zhu, 2002) both at training and test time. The BLEU Score matches the presence of the exact tokens in the source and reference document.

4.2.2 Evaluation Data Sets

The following data sets are used for evaluation.

1. **WMT-News**
   This part of the evaluation data set represents how close our system is to a general-purpose translation system. The WMT-News data set is a different domain from the customer data set and hence is a good verification step against overfitting. The entire WMT-News data set of 14K sentences is used here. All the text in this data set is converted to lowercase to test if the system is robust against wrong casing.

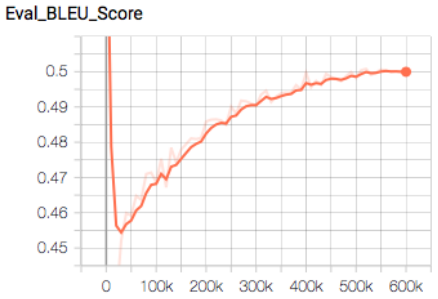
2. **Customer Correspondence Letters**
   The customer correspondence letters, part 2, comprising of 141,018 sentences are used here. These are sentences from in-domain letters that the customer often uses.

5 Experiments and Results

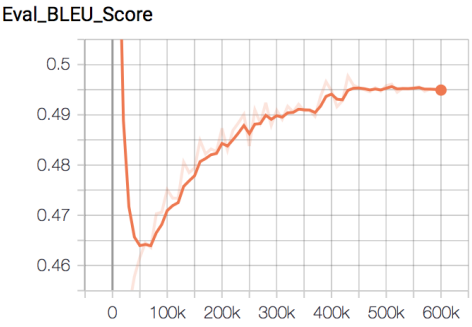
The following experiments are done on the Transformer model[2] with fixed training hyperparameters using OpenSeq2Seq (Ginsburg et al., 2018).

1. **Experiment 1: Reference Baseline**
   First, we trained a model from only the raw public data set (without any preprocessing steps) and tested it on the two evaluation sets. This model is considered as our reference baseline.

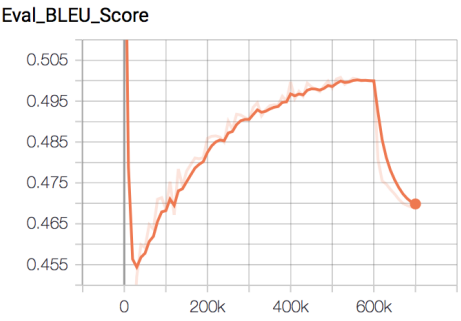
[2]https://nvidia.github.io/OpenSeq2Seq/html/machine-translation/transformer.html



(a) Experiment 1: Reference Baseline



(b) Experiment 2: Clean and Augmented Data



(c) Experiment 3: Fine-tuning on the Customer Data

Figure (1) BLEU score at each training step

2. **Experiment 2: Clean and Augmented Data**
   We use the alignment tool by YASA (Lamraoui, Langlais, 2013) to align sentences. Then we eliminate address lines, augment the data with random punctuation and casing. This reduces the amount of training data but enhances the quality of it.

3. **Experiment 3: Fine-tuning on the Customer Data**
   The model from experiment 2 performed better on the customer evaluation data set. Hence, it was chosen for fine-tuning on the customer correspondence letters (part 1). The attention dropout parameter is slightly increased to protect against overfitting.

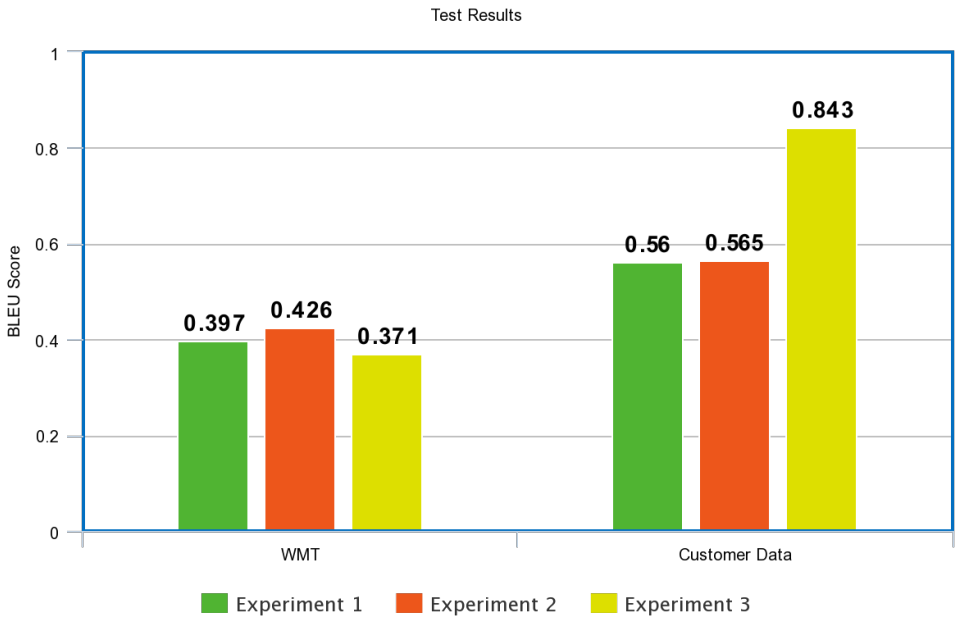


Figure (2) BLEU scores on evaluation data sets

Figure 1 shows the BLEU score evaluation graphs during training for all the above mentioned experiments. We can see that at train time experiments 1 and 2 perform almost identically and in experiment 3, the BLEU score drops slightly in the fine-tuning phase. This is due to the domain difference between the public and customer data sets. Note that the language model in experiment 3 is the same as experiment 2.

The BLEU scores on the test data for these experiments are shown in figure 2. As we can see with 38M sentences, the baseline reference performs well but falls short in the customer domain with a BLEU score of 0.560 on the customer evaluation data set. Also, this model trained only on raw public data does not perform as well on WMT-News lowercase.

Experiment 2, which involves the aligned, cleaned, and augmented data, starts improving on WMT-News but, it is still mediocre on the customer data set. This further validates the argument that the translator trained on a general data set can not cater to domain-specific needs.

Experiment 3 involves fine-tuning the model from experiment 2 on the customer correspondence letters (part 1). While it shows no improvement at train time, the BLEU score on the in-domain evaluation data set improves greatly and reaches ***0.8****.*

6 NMT Inference Service Deployment

We built a scalable, performance-oriented, and cost-optimized deployment pipeline targeting a cloud-native environment. We separated all text processing from neural model inference. Text processing and clients serving rest APIs were implemented as light-weight microservices that run on CPU. Neural models are served by TensorRT inference Server containers (Nvidia, 2019), which are provisioned with GPU. Models are placed on persistent storage accessible to the TensorRT Inference Servers (Figure 3).

We chose TensorRT inference server because of the following features that it provides:

- Concurrent model execution
  Since TensorRT can access multiple models or multiple instances of the same model at the same time, it can be decided at run time which model will be used for inference.

- Seamless model deployment
  Models are stored in the file system-based model repository. Each model is represented by a directory. This directory contains a model configuration file that describes the framework, scheduling, batching, concurrency, and other model serving parameters. Each model can have one or more versions available in the model repository. Each version is stored in its own, numerically named

subdirectory where the name of the subdirectory corresponds to the version number of the model. The server monitors all changes in the model store and adds or removes models or model versions from serving without any restarts of TensorRT Inference Server.

- Batching support
  It provides multiple batching and scheduling algorithms that combine individual inference requests to improve inference throughput.
- Optimized models
  - Layer and tensor fusion and elimination of unused layers
  - Precision Calibration (support for FP16 and INT8 precision)
  - Kernel Auto-tuning
  - Efficient memory reuse
- Scalable and reliable deployment
  Since the model serving and processing of text are independent of each other and model serving is dynamic, a new version of the model can be deployed without a server restart or any downtime in the service. Since multiple versions are present, rollback to a previous version is easy to implement.
- Extensible Architecture
- Inference and server monitoring API

Utilizing a TensorRT inference server decreases the inference response time by a factor of three, due to the use of optimized models and GPU sharing. Our approach also allows a separate autoscaling of CPU and GPU resources.

6.1 Model Deployment Pipeline

Due to the independence of the neural model serving and pre/post-processing, the model can be deployed and rolled back without rebuilding the images and restarting TensorRT Inference Server containers – all that is needed to be done is to change environment variables and restart CPU microservices. Here is how it is achieved.

1. A Git repository contains the code of the service and the configuration YAML of the model deployed. It also contains a neural model metadata file. Model metadata file includes parameters like the corresponding language model location, output, input tensor name, and model version.

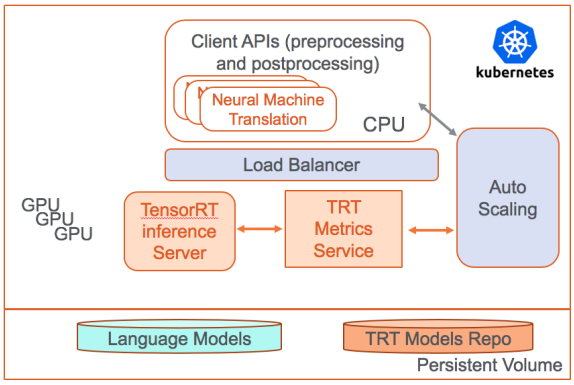


Figure (3) Current architecture of Translation system deployed.

2. Anytime a new neural model is trained, and subsequently the parameters of the metadata are changed and committed, a minimal testing docker container is automatically created. The new language model and inference model are deployed to the persistent volume storage, accessible to the TensorRT Inference Server and text processing containers. TensorRT Inference Server automatically starts serving a new version of the model in addition to the current one. General sentences are passed through the model to check its sanity. The deployment only goes further if the model translates these source language phrases correctly.

3. Depending upon which customer/use case the model will serve, the model is then evaluated on sentences of a specific domain.

4. If the aforementioned steps are successful all that is needed to be done is environment variable changes for text processing (TensorRT client) container. The rolling update with the new environment variables is initiated. After this update, APIs are serving the new model.

6.2 Online Learning

The API lets the customer upload a document and then shows all the parts of the document and corresponding translated parts simultaneously. This allows the enterprise users to edit the translations if they want any modification in the translated document. The customer can download the modified document containing the changes they made.
These corrections are being recorded so that the model could be improved periodically. How to judge the accuracy of customer corrections and

how to use that information for online learning makes a top priority in our future work.

7 Conclusion

Our experiments demonstrate how a general-purpose neural machine translation framework can be customized to a specific use case for a specialized domain enterprise. They also show how different versions of the same model architecture can serve different needs. For instance, experiment 2 yields a model that is suitable for a manually typed general language data but not suitable for medical claim correspondence letters for the customer. Experiment 3, however, yields a model that performs exceedingly well in the given customer scenario and has a BLEU score of 0.8 which would be very hard to manifest for a general-purpose translator.
In this work, we also describe an architecture for the deployment of deep learning models (specifically neural machine translation) optimized for inference using TensorRT. We explain how the models can be automatically deployed and changed at run time following the customer's needs. For instance, the translation model of experiment 2 is served for a human typing interface, whereas the fine-tuned model from experiment 3 is provided for medical correspondence letters.

Acknowledgements: We would like to thank our colleagues Ajay Ajit Maity, Brian Carter and Declan Atkins for their contribution.

References

Cho et al. 2014. *Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation* Prentice-Hall, Englewood Cliffs, NJ.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. 1997. *Neural computation*. 9(8):1735–1780, 1997.

Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural networks. 2014. *Advances in Neural Information Processing Systems*, 3104–3112, 2014. (2)

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 1981. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014. (1).

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*

Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. 2016. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*

Ankur Parikh, Oscar Tackstrom, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model. *Empirical Methods in Natural Language Processing*

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130, 2017.*

Fethi Lamraoui, Philippe Langlais, 2013. Yet Another Fast, Robust and Open Source Sentence Aligner. Time to Reconsider Sentence Alignment? *Machine Translation summit, 2013*

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu 2002. BLEU: a Method for Automatic Evaluation of Machine Translation *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, July 2002, pp. 311-318. Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu*

Oleksii Kuchaiev and Boris Ginsburg and Igor Gitman and Vitaly Lavrukhin and Jason Li and Huyen Nguyen and Carl Case and Paulius Micikevicius 2018. Mixed-Precision Training for NLP and Speech Recognition with OpenSeq2Seq *arXiv 1805.10387 cs.CL*

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin 2017. Attention Is All You Need *arXiv 1706.03762cs.CL*

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin 2012. Parallel Data, Tools and Interfaces in OPUS *In Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012)*

Chu, Chenhui and Wang, Rui 2018. A Survey of Domain Adaptation for Neural Machine Translation *Proceedings of the 27th International Conference on Computational Linguistics*

Nvidia-TensorRT 2019. `https://github.com/NVIDIA/TensorRT`