Towards Eco-Friendly, Efficient & Effective Edge-Device Neural Machine Translation

Dane Williamson
University of Virginia (UVA)
Charlottesville, VA
dw3zn@virginia.edu

ABSTRACT

UPDATED—May 5, 2021. This paper seeks to outline the economic, environmental and computational costs associated with Neural Translation Models (NMT) within Cyber-Physical Systems (CPS) and the Internet of Things (IoT). While these costs are too vast and numerous to be immediately resolved within this paper, a precedent is set for augmented development and deployment pipelines to reduce the computational costs and carbon footprint of ML Inference by performing computation on models stored on Edge Computing (EC) Devices.

Author Keywords

Neural Machine Translation; Mobile Devices; Open Neural Network Exchange; Edge Computing; Attention

CCS Concepts

•**Human-centered computing** → Ambient intelligence; •**Computing methodologies** → **Machine translation;** •**General and reference** → General conference proceedings; •**Hardware** → *Impact on the environment;*

INTRODUCTION

Training state-of-the art Neural Machine Translation (NMT) models on modern datasets is often computationally expensive. Even with the usage of multiple computers with Graphics Processing Units (GPUs), the training process can take days. The model produced is typically of the order of several Gigabytes (GB) in size and is thus unsuitable for usage on a typical EC device as the typical EC device has only a few GB of storage. The Switch Transformer model (Switch-C) [7] for example, has approximately over a trillion parameters and is trained on a 745 GB dataset. The Switch transformer holds record results on multiple Natural Language Processing (NLP) benchmarks. Cyber-Physical systems which are interactive with humans will no doubt be unable to provide the resources required to support such a model. Thusly, inference is typically offloaded to a cloud device which can provide these resources. This offloading introduces additional latency and entirely negates the effectiveness of devices when network connectivity is less than optimal or entirely compromised or unavailable.

NMT has achieved stellar performance in large-scale, high-resource environments. Since these resources are typically unavailable in the pocket of an average user or inside the average IoT device and the need for translation is ever more required in a diverse and increasingly connected world, in this paper a compromise is sought for model resources and translation accuracy. Using various modern techniques and frameworks a proof of concept for a low resource, computationally inexpensive, high accuracy NMT model for deployment on edge-computing devices is provided.

The contributions made in this paper are as follows:

- The need for considering the financial and environmental impact of computationally expensive machine learning architectures is demonstrated.

- A precedent is provided for ensuring that computationally expensive models can be deployed on less computationally powerful devices for Cyber Physical Systems and the Internet of Things.

- The carbon footprint of inferences made by a Neural Machine translation system is reduced by ensuring inference can be performed without the utilization of cloud resources.

The utility of these contributions is demonstrated by performing an NMT task of converting English to French.

BACKGROUND AND MOTIVATION

NMT may potentially address the shortcomings of traditional machine translation systems, such as Statistical Machine Translation (SMT), due to its ability to directly learn the associations between an input text and output text. Despite the merits of NMT, it still has inherent deficiencies such as slow training times, slow inference speed and fragility to rare or out of vocabulary words. For deployment on resource-constrained edge devices, NMT models are entirely unsuited. Modern translation systems such as google translate are ubiquitously available to users with an internet connection. However in

CHI'21, May 25–30, 2021, Honolulu, HI, USA

ISBN 978-1-4503-6708-0/21/05. . . $15.00

DOI: https://doi.org/10.1145/3313831.XXXXXXX

situations where translation is required but connectivity is limited, this is inconvenient. Interactive Cyber-Physical Systems should be well-equipped to be responsive to a user regardless of his or her language of choice. To address this issue the deployment of pre-trained models on mobile devices is proposed. The inference time and memory occupancy of these models is not suitable for immediate deployment on typical edge devices. However model compression can reduce the memory occupancy and inference time of these devices and render them more efficient.

Natural Language Processing in Cyber-Physical Systems and the Internet of Things

A Cyber-Physical system may be formally defined as a system that combines the capabilities of computing, communications and data storage to monitor or control entities in the physical world. [15] Recent research in the field of CPS has seen application of NLP techniques to optimize human-in-the-loop CPS. In [11] researchers leverage Language Modeling techniques for extracting Signal Temporal Logic (STL) specifications for smart cities. In [2] researchers leverage NLP for integration in home automation using chat bots.

In an increasingly interconnected and nomadic world, society is communicating at an unprecedented rate. Interactive CPS will no doubt be required to interface with individuals from a variety of linguistic backgrounds. The additional overhead of performing translational inference on cloud devices will degrade the performance of these interactive CPS. Furthermore the cloud-based operations will create a broader attack surface to compromise these systems. The work proposed here therefore seeks to provide a compromise for computationally inexpensive edge-based NMT for Interactive CPS.

The aforementioned desire for Edge-Device based Inference in Interactive CPS and AI-enabled IoT devices is an area of research that has recently received attention. In the forthcoming IoT revolution, there will no doubt be various sources of network latency and bottleneck as devices seek to utilize network bandwidth. In [13], the researchers seek to pave the way to the "Internet of Conscious Things" by reviewing the main techniques for executing ML inference on low-performance hardware. A noteworthy finding in [13] is that Neural Network based machine learning typically requires less computational power in deployment than in training. For ML inference in the Internet of Conscious Things, [13] propose several solutions such as lightweight model design, model compression and optimal edge computing hardware. In light of these proposed solutions, the nature of the typical NLP model must be considered before a solution for deployment on edge devices can be synthesized.

Resource Consumption

Modern NLP has seen an abundance of language models which involve a large number of parameters and training data. [4] The advent of modern deep-learning systems has not come without a significant trade-off between model performance and environmental impact. Recent research has been focused in conflicting and opposite directions, i.e. a line of research has been focused on building relatively large models with many parameters designed for computationally complex tasks, see Table 1. By contrast other researchers have aimed to build models which are more computationally efficient. In [8], the researchers propose online leaderboards to motivate and incentivize research in energy efficient and computationally effective machine learning (ML) models. As stated by [8] a major roadblock to a comprehensive understanding of the potential climate impacts of ML research is a lack of resources to track the realtime energy consumption and carbon emissions associated with ML research and work particularly in fueling modern tensor processing hardware.

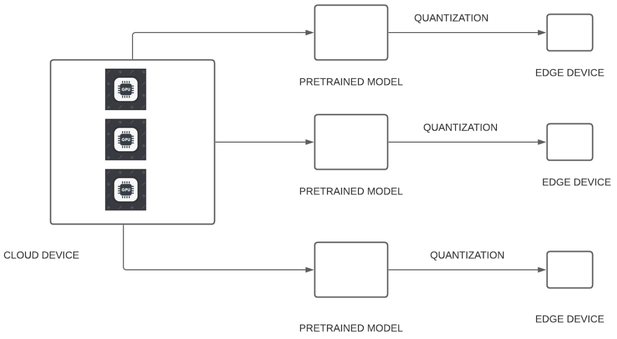

Figure 1. NMT Edge Deployment Framework

While researchers such as [8] seek to establish a centralised framework for reporting the carbon impact of ML research, others such as [19] seek to estimate the financial and environmental costs of training successful modern neural network architectures. In fact, an astounding claim made in [19] is that the model emissions from training the popular transformer, BERT [6] are roughly equivalent to a those of a trans-American flight. To achieve marginally higher state-of-the-art Bilingual Evaluation Understudy (BLEU) [14] scores, researchers potentially incur several hundreds of thousands of dollars worth of cost in computation power and produce non-trivial carbon emissions. As if to underscore this issue, Strubel et al [19] implore the NLP community to prioritize the development of efficient models and hardware.

The issue of excessive resource requirements for NLP models on resource constrained edge devices is clearly illustrated. In [21], a system for compressing deep neural networks for sensing applications on embedded devices is developed, called DeepIoT. In following the precedent set by DeepIot, a framework for compressing deep neural networks for NMT on edge devices is developed here. Whereas the researchers in [13] developed a novel algorithm for compression, the popular compression method of quantization is used here. The framework produced can be visualised in Figure 1.

Model Compression

The aim of model compression of deep neural networks is to produce a simplified model while not significantly reducing accuracy. The term simplified here refers to a model which has fewer parameters and will use less memory (RAM) during inference. A simplified model is also likely to see a reduction

Year	Model	# of Parameters	Dataset Size
2019	BERT	3.4E+08	16GB
2019	DistillBERT	6.6E+07	16GB
2019	ALBERT	2.23E+08	16GB
2019	XLNet (Large)	3.40 E+08	126 GB
2020	ERNIE-GEN (Large)	3.40 E+08	16 GB
2020	GPT-3	1.75E+11	570 GB
2021	Switch-C	1.57E+12	745 GB

Table 1. Overview of Recent Large Language Models [4]

in inference time and lower energy consumption at inference. There are several popular methods of modular compression such as: (i) Pruning, (ii) Quantization, (iii) Low-rank approximation and sparsity, (iv) Knowledge distillation, (v) Neural Architecture Search.

Pruning

Network pruning is essentially the removal of parameters that do not have an impact on network inference accuracy. [10] A parameter or neuron may be considered redundant if the coefficients of the weight values are close to zero or exactly zero.

Knowledge Distillation

In knowledge distillation (KD) the aim is to produce a simplified model which is able to match the inference accuracy of a more complex model. In KD a larger model is used to train a less computationally complex student network which imitates the functions of the larger model. [10]

Network Architecture Search

Network architecture search is a method of algorithmically searching for an efficient network from a predetermined search space.

Quantization

Quantization is the process of approximating a continuous signal by a set of discrete symbols or integer values. [10] The goal of model quantization is essentially to reduce the precision of the network components but only to the extent that there is a noticeable reduction in inference time but not in accuracy.

Static Quantization

Static Quantization improves the latency of networks by quantizing the weights ahead of inference. The scale factor and bias for the activation tensors are based on observing the behaviour of the model during a calibration process.

Dynamic Quantization

Dyniamic Quantization improves network latency by quantizing the weights before inference, however the activations are dynamically quantized during inference. Dynamic Quantization is more appropriate in situations where the model execution time is dominated by loading weights from memory and not tensor computations. As the NMT model developed consists of two Gated Recurrent Units (GRUs) [5], dynamic quantization is more appropriate.

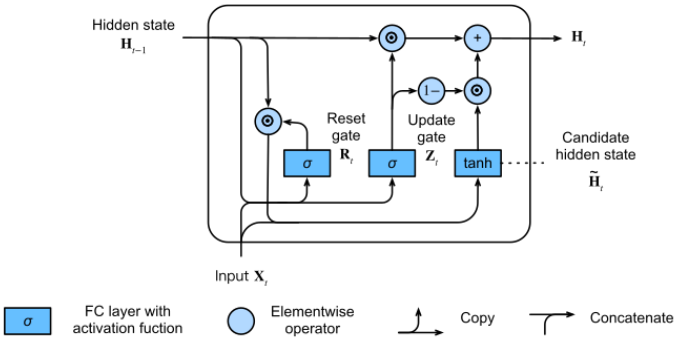

Figure 2. Illustration of a Gated Recurrent Unit Cell Architecture [5]

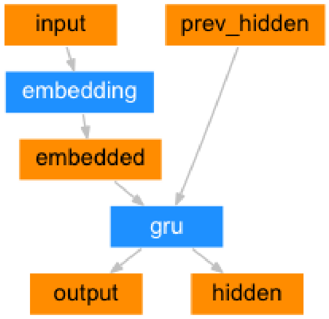

Figure 3. Encoder Network Architecture

NEURAL TRANSLATION MODEL

Neural Machine Translation consists of utilizing a neural network model to transform a source sentence expressed in one language to a target sentence expressed in a different language. State-of-the-art translation models typically consist of a Sequence-to-Sequence (Seq2Seq) structure. This structure as outlined in [20] consists of a (GRU) cell (see Figure 2) to map the input sequence into a vector. A separate GRU cell is used to decode the target sequence from the encoded vector. To implement the translation model, the steps outlined in the official pytorch tutorial are synthesized. [16] The specific translation task being performed by the model in this experiment is that of translating an English word, phrase, saying or sentence into its French equivalent.

Encoder

The encoder model is a multi-layer GRU Recurrent Neural Network (RNN) which will take as input a single word from the input sentence. At each time step, the encoder will produce a vector representation for each word as well as a hidden state to be provided as input for the next word in the sequence. The hidden state for the initial token in the sequence is a vector initialized with zeros. For further information on the GRU functionality see Figure 2. The final state output by the encoder is used as the initial hidden state of the decoder. This last output has encoded context from the entire sequence. The encoder architecture is shown in Figure 3.

Decoder

The decoder takes as input for the first hidden state the final context vector produced from the encoder. The first

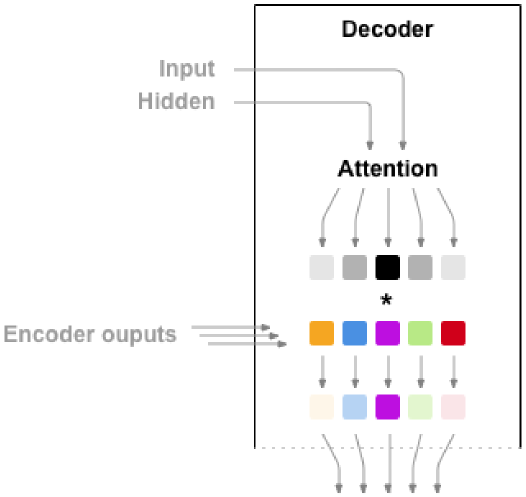

Figure 4. Attention Mechanism

token used in generating the sequence is the start of sentence '<sos>' token. The global attention mechanism used in [3] is leveraged to focus on salient parts of the input sequence and thus learn the alignment between the segments of the output sequence and their corresponding segments in the input sequence. The decoder's input and hidden state are used to calculate the attention weights for the decoder. These weights are then multiplied by the encoder output vectors to produce a weighted combination. This weighted combination assists the decoder in producing the right output words. The attention mechanism is illustrated in Figure 4. To increase the rate at which the model converges, teacher forcing is employed to occasionally directly feed the target outputs as the input at each next time step instead of the decoder's own output. The decoder architecture is shown in Figure 5.

TRAINING

The training process used in the official pytorch tutorial, [16] is replicated here to create the multi-task NMT model outlined in [17]. The dataset used to train the models is taken from the Tatoeba online repository of example sentences for foreign-language learners. The language pair source sentences and translations are also made available. Each token in the sentence is replaced with it's index representation in the vocabulary which is manually generated by assigning an integer value to be associated with each token. An end of sentence ('<eos>') token is appended to the sentence before conversion to a tensor. The hidden size of the encoder and decoder is set to 512 and the training process takes place for 75000 epochs. Model training is done on the publicly available Google Colab GPUs.

To track the loss incurred by the encoder in producing the vectorized representation and the decoder in producing the translations, the Negative Log Likelihood Loss function is used. This is possible as determining if representations are correct is essentially a classification problem between the representations produced by the model and the labelled output translations. The Stochastic Gradient Descent [18] (SGD) Optimization algorithm is utilized for model convergence. A teacher forcing ratio of 0.5 was used to ensure the equal likelihood of the input of the next time step of the decoder being either the true label or the previous decoder output.

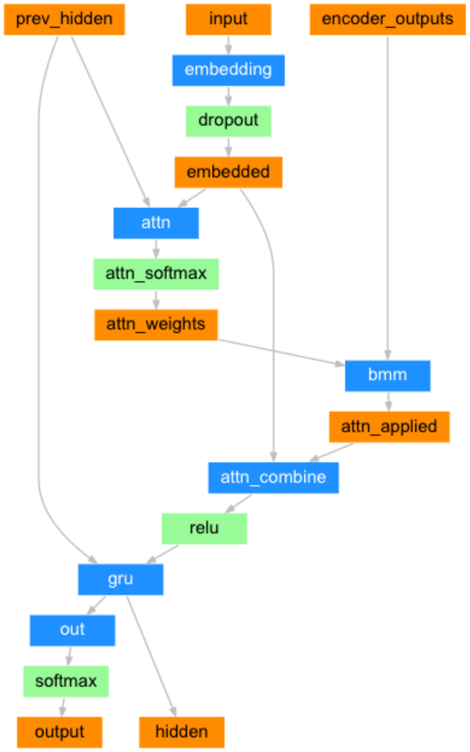

Figure 5. Decoder Network Architecture

EVALUATION

The BLEU score [14] of the nmt model was evaluated using the available pytorch and nltk libraries. Formally, the BLEU score is defined by [1] as:

$$BLEU = \underbrace{min(1, exp(1 - \frac{reference\_length}{output\_length}))}_{brevity\ penalty} \underbrace{(\prod_{i=1}^{4} precision_i)^{\frac{1}{4}}}_{n-gram\ overlap}$$

with:

$$precision_i = \frac{\sum_{snt \in CandCorpus} \sum_{i \in snt} min(m_i cand, m_i ref)}{w_t^i = \sum_{snt' \in CandCorpus}}$$

where

- m_{cand}^i is the count of i-gram in the candidate sentence matching the reference translation.
- m_{ref}^i is the count of i-gram in the reference translation.
- w_t^i is the total number of i-grams in candidate translation.
- The **Brevity Penalty** penalizes generated translations that are too short compared to the closest reference length.

English-French Translations

src	you re temperamental .
ref	tu es fantasque .
translation	vous etes lunatiques .
src	i am afraid to go .
ref	j ai peur de m y rendre .
translation	j ai peur d y aller .
src	you re the oldest .
ref	tu es la plus vieille .
translation	c est le plus vieux .
src	i m surprised you didn t know that .
ref	je suis surpris que vous ignoriez cela .
translation	je suis surpris que vous ignoriez cela .

Table 2. Sample Translations

- The **N-Gram Overlap** counts how many n-grams in the candidate match their n-gram counterpart in the reference translation.

Sample translations from the quantized and original model are shown in table 2. In lieu of a holdout dataset for evaluation, the BLEU score of the NMT model on 5000 samples of the corpus of English-French sentence pairs is determined to be approximately 21. When the encoder and decoder models are quantized, there is no reduction in BLEU score. Indicating that satisfactory ML inference can be performed within embedded systems and edge devices.

QUANTIZATION PROCESS

The weight values of the models produced were determined to be approximately 12 MegaBytes (MB) for the encoder and approximately 26 MegaBytes (MB) for the decoder.

To reduce the inference time of the neural translation model, dynamic quantization is performed using the beta-stage pytorch libraries available. The evaluation results for the quantized and original models can be seen in Table 4. While the encoder model saw a sizeable reduction in size from 12 MB to 7 MB, there was no reduction in inference accuracy. In the case of the decoder however, the model size was reduced to a half of the original size, also with no loss in inference accuracy.

MODEL CONVERSION

The Open Neural Network Exchange (ONNX) standard is an open-source machine independent format that is used for interoperability and exchange between neural network models. [9] Pytorch has inherent support for exporting models into the ONNX format and operating them with the ONNX runtime. Unfortunately at the time of this writing, Pytorch does not have support for converting **quantized** models to ONNX format.

CODE

The code used in this project can be found in the GitHub repository. The notebook for experiments, code for the Encoder and Decoder GRU models and the weight files for both the original quantized weights are available.

Bleu Score	Translation Quality
<10	Almost Useless
10 - 19	Hard to get the gist
20-29	Gist is clear w Significant Grammatical Errors
30 - 40	Understandable to good translations
40-50	High Quality
50 - 60	Very High Quality, Adequate, Fluent
>60	Better than human

Table 3. Bleu Score Scale

Model	Size (MB)	BLEU Score
Quantized NMT	7 (Encoder) 13 (Decoder)	21
Original Torch Model	12 (Encoder) 26 (Decoder)	21

Table 4. Table Showing BLEU Score for Quantized and original models

LIMITATIONS

The work done in this paper is in response to the actionable recommendations proposed in [19] to reduce the environmental impact and financial and computational costs associated with NLP model training and inference. A translation model of satisfactory BLEU score was developed and quantized to occupy less memory and reduce inference time, with no loss of accuracy. However due to the lack of support for conversion of quantized models into onnx format, these models could not be converted into various formats, especially those suitable for usage on edge devices and embedded systems.

FUTURE WORK

Extensions for the work done in this paper include incorporating a Federated Learning (FL) framework for ML inferences. Federated learning is a distributed machine learning paradigm introduced by Google which aims to enable edge-devices to collaboratively learn a shared ML model without sharing private or sensitive data. [12]. Current FL frameworks assume heterogeneity within their respective aggregation schemes, which may be difficult to realise given the varied nature of embedded systems and edge devices. However FL still provides a promising avenue of researching energy efficient deep learning.

The development of an Application Programming Interface (API) to support the conversion of Quantized Pytorch models to Open Neural Network Exchange Format is also an avenue for extended work which is being considered. The lack of support for conversion of models from a quantized pytorch framework to an ONNX framework underscores the current underestimation of the need for model efficiency in the ML community.

REFERENCES

[1] Evaluating models nbsp;|nbsp; AutoML Translation Documentation nbsp;|nbsp; Google Cloud. (????). https://cloud.google.com/translate/automl/docs/evaluate

[2] Cyril Joe Baby, Faizan Ayyub Khan, and J. N. Swathi. 2017. Home automation using IoT and a chatbot using natural language processing. In *2017 Innovations in Power and Advanced Computing Technologies (i-PACT)*. 1–6. DOI: http://dx.doi.org/10.1109/IPACT.2017.8245185

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural Machine Translation by Jointly Learning to Align and Translate. (2016).

[4] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? . In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (FAccT '21)*. Association for Computing Machinery, New York, NY, USA, 610–623. DOI: http://dx.doi.org/10.1145/3442188.3445922

[5] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *CoRR* abs/1412.3555 (2014). http://arxiv.org/abs/1412.3555

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (2019).

[7] William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. (2021).

[8] Peter Henderson, Jieru Hu, Joshua Romoff, Emma Brunskill, Dan Jurafsky, and Joelle Pineau. 2020. Towards the Systematic Reporting of the Energy and Carbon Footprints of Machine Learning. (2020).

[9] Tian Jin, Gheorghe-Teodor Bercea, Tung D. Le, Tong Chen, Gong Su, Haruki Imai, Yasushi Negishi, Anh Leu, Kevin O'Brien, Kiyokuni Kawachiya, and Alexandre E. Eichenberger. 2020. Compiling ONNX Neural Network Models Using MLIR. (2020).

[10] Tailin Liang, John Glossner, Lei Wang, and Shaobo Shi. 2021. Pruning and Quantization for Deep Neural Network Acceleration: A Survey. (2021).

[11] Meiyi Ma, John A. Stankovic, and Lu Feng. 2018. Cityresolver: A Decision Support System for Conflict Resolution in Smart Cities. In *Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS '18)*. IEEE Press, 55–64. DOI: http://dx.doi.org/10.1109/ICCPS.2018.00014

[12] H. Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. 2016. Federated Learning of Deep Networks using Model Averaging. *CoRR* abs/1602.05629 (2016). http://arxiv.org/abs/1602.05629

[13] Massimo Merenda, Carlo Porcaro, and Demetrio Iero. 2020. Edge Machine Learning for AI-Enabled IoT Devices: A Review. *Sensors* 20, 9 (2020). DOI: http://dx.doi.org/10.3390/s20092533

[14] Kishore Papineni, S. Roukos, T. Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *ACL*.

[15] Ricky Henry Rawung and Aji Gautama Putrada. 2014. Cyber physical system: Paper survey. In *2014 International Conference on ICT For Smart Society (ICISS)*. 273–278. DOI: http://dx.doi.org/10.1109/ICTSS.2014.7013187

[16] Sean Robertson. NLP From Scratch: Translation with a Sequence to Sequence Network and Attention¶. (????). https://pytorch.org/tutorials/intermediate/seq2seq_translation_tutorial.html#the-seq2seq-model

[17] Raphael Rubino, Benjamin Marie, Raj Dabre, Atsushi Fujita, Masao Utiyama, and Eiichiro Sumita. 2020. Extremely low-resource neural machine translation for Asian languages. *Machine Translation* 34 (12 2020). DOI:http://dx.doi.org/10.1007/s10590-020-09258-6

[18] Sebastian Ruder. 2017. An overview of gradient descent optimization algorithms. (2017).

[19] Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and Policy Considerations for Deep Learning in NLP. *CoRR* abs/1906.02243 (2019). http://arxiv.org/abs/1906.02243

[20] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. (2014).

[21] Shuochao Yao, Yiran Zhao, Aston Zhang, Lu Su, and Tarek F. Abdelzaher. 2017. Compressing Deep Neural Network Structures for Sensing Systems with a Compressor-Critic Framework. *CoRR* abs/1706.01215 (2017). http://arxiv.org/abs/1706.01215