# SOCKEYE 2:
# A Toolkit for Neural Machine Translation

**Felix Hieber** and **Tobias Domhan** and **Michael Denkowski** and **David Vilar**
Amazon
{fhieber,domhant,mdenkows,dvilar}@amazon.com

## Abstract

We present SOCKEYE 2, a modernized and streamlined version of the SOCKEYE neural machine translation (NMT) toolkit. New features include a simplified code base through the use of MXNet's Gluon API, a focus on state-of-the-art model architectures, and distributed mixed precision training. These improvements result in faster training and inference, higher automatic metric scores, and a shorter path from research to production.

## 1 Introduction

SOCKEYE (Hieber et al., 2017) is a versatile toolkit for research in the fast-moving field of NMT. Since the initial release, it has been used in at least 25 scientific publications, including winning submissions to WMT evaluations (Schamper et al., 2018). Based on the deep learning library MXNet (Chen et al., 2015), SOCKEYE also powers Amazon Translate, showing industrial-strength performance in addition to the flexibility needed in academic environments. Moreover, we are excited to see hardware manufacturers contributing optimizations to MXNet and SOCKEYE. Intel has demonstrated large performance gains for SOCKEYE inference on Intel Skylake processors.[1] NVIDIA is working on significant performance improvements for SOCKEYE's transformer (Vaswani et al., 2017) implementation through fused operators and optimized beam search. This paper discusses SOCKEYE 2's streamlined Gluon implementation (§2), support for state of the art architectures (§3), and improved model training (§4).

## 2 Gluon Implementation

SOCKEYE 2 adopts Gluon, the latest and preferred application programming interface (API) of the MXNet deep learning library. Gluon simplifies the code while improving overall performance. Developers can define building blocks of neural network architectures as Python classes and seamlessly switch between eager execution for step-by-step debugging and cached computation graphs for maximum performance. Migration to Gluon simplifies training and inference code in SOCKEYE 2, reducing the overall number of lines of Python code by 25%. The hybridized Gluon transformer implementation in SOCKEYE 2 improves training speed by 14%, compared to SOCKEYE.

## 3 Focus on State-of-the-Art Models

Due to the success of self-attentional models we concentrated development of SOCKEYE 2 on the transformer (Vaswani et al., 2017), removing support for recurrent and convolutional models. Using the pre-norm configuration by default allows for learning rate warm-up-free training.

We found deep encoders and shallow decoders for transformers to be competitive in BLEU while significantly increasing decoding speed due to computational workload being shifted to the encoder side. On WMT19 (EN–FI, FI–EN, EN–DE, DE–EN), a 20-encoder and 2-decoder layer configuration improves on average by 0.2 BLEU over the baseline, while reducing decoding time by 50%.

We also improved support for source factors by allowing to tie source factor and word embeddings,

[1] https://www.intel.ai/amazing-inference-performance-with-intel-xeon-scalable-processors/#gs.wrgsji

|  | DE–EN |  | EN–FI |  |
|---|---|---|---|---|
|  | BLEU | Time | BLEU | Time |
| Ott et al. (2018) | 34.7 | 30h | 20.1 | 14h |
| Plateau Reduce | **34.9** | **28h** | **20.7** | **12h** |

**Table 1:** SacreBLEU (Post, 2018) scores (newstest2019) and training times (8 NVIDIA V100 GPUs) for a 20-encoder 2-decoder layer transformer using the training setup described by Ott et al. (2018) and Plateau Reduce, both implemented in SOCKEYE 2.

as well as specifying different types of embedding combinations (concatenation or summation).

## 4 Training Improvements

SOCKEYE 2 significantly accelerates training with Horovod[2] integration (Sergeev and Balso, 2018) and MXNet's automatic mixed precision (AMP). Horovod extends synchronous training to any number of GPUs (including across nodes) while AMP automatically detects and converts parts of the model that can run in reduced-precision mode (FP16) without loss of quality.

SOCKEYE also provides a data-driven alternative to the popular "inverse square root" learning schedule used by Vaswani et al. (2017) and Ott et al. (2018): "Plateau Reduce" keeps the same learning rate until validation perplexity does not increase for several checkpoints, at which time it reduces the learning rate and rewinds all model and optimizer parameters to the best previous point. Training concludes when validation perplexity reaches an extended plateau. In a WMT19 benchmark (Barrault et al., 2019), Plateau Reduce training produces stronger models in slightly less time than the setup described by Ott et al. (2018). Results are presented in Table 1 where all values are averages over 3 independent training runs with different random initializations. Full hyperparameters for SOCKEYE 2's large batch training can be found in the toolkit's documentation.

## 5 Licensing and availability

SOCKEYE 2 is available[3] under the Apache 2.0 license. It includes a Docker build to easily run training or inference on any supported platform.

## 6 Conclusion

SOCKEYE 2 provides out-of-the-box support for quickly training strong transformer models for research or production. Extensive configuration options and the simplified code base enable rapid development and experimentation. We invite the community to contribute their ideas to SOCKEYE 2 and hope that the new programming model and performance improvements enable others to conduct effective and successful research.

[2] https://github.com/horovod/horovod

[3] https://github.com/awslabs/sockeye

## References

Barrault, Loïc, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 conference on machine translation (WMT19). In *Procs. of the Fourth Conference on Machine Translation (Vol. 2: Shared Task Papers)*, pages 1–61, Florence, Italy, August.

Chen, Tianqi, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. 2015. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*.

Hieber, Felix, Tobias Domhan, Michael Denkowski, David Vilar, Artem Sokolov, Ann Clifton, and Matt Post. 2017. Sockeye: A toolkit for neural machine translation. *ArXiv e-prints*, abs/1712.05690.

Ott, Myle, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Procs. of the Third Conference on Machine Translation, Vol. 1: Research Papers*, pages 1–9, Belgium, Brussels, October.

Post, Matt. 2018. A call for clarity in reporting bleu scores. In *Procs. of the Third Conference on Machine Translation, Vol. 1: Research Papers*, pages 186–191, Belgium, Brussels, October.

Schamper, Julian, Jan Rosendahl, Parnia Bahar, Yunsu Kim, Arne Nix, and Hermann Ney. 2018. The RWTH Aachen University supervised machine translation systems for WMT 2018. In *Procs. of the Third Conference on Machine Translation, Vol. 2: Shared Task Papers*, pages 500–507, Belgium, Brussels, October.

Sergeev, Alexander and Mike Del Balso. 2018. Horovod: fast and easy distributed deep learning in tensorflow. *CoRR*, abs/1802.05799.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.