Findings of the Third Workshop on Neural Generation and Translation

Hiroaki Hayashi[♢], **Yusuke Oda**[♣], **Alexandra Birch**[♠], **Ioannis Konstas**[△], **Andrew Finch**[♡], **Minh-Thang Luong**[♣], **Graham Neubig**[♢], **Katsuhito Sudoh**[⋆]

[♢]Carnegie Mellon University, [♣]Google Brain, [♠]University of Edinburgh
[△]Heriot-Watt University, [♡]Apple, [⋆]Nara Institute of Science and Technology

Abstract

This document describes the findings of the Third Workshop on Neural Generation and Translation, held in concert with the annual conference of the Empirical Methods in Natural Language Processing (EMNLP 2019). First, we summarize the research trends of papers presented in the proceedings. Second, we describe the results of the two shared tasks 1) efficient neural machine translation (NMT) where participants were tasked with creating NMT systems that are both accurate and efficient, and 2) document generation and translation (DGT) where participants were tasked with developing systems that generate summaries from structured data, potentially with assistance from text in another language.

1 Introduction

Neural sequence to sequence models (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014; Bahdanau et al., 2015) are now a workhorse behind a wide variety of different natural language processing tasks such as machine translation, generation, summarization and simplification. The 3rd Workshop on Neural Machine Translation and Generation (WNGT 2019) provided a forum for research in applications of neural models to machine translation and other language generation tasks (including summarization (Rush et al., 2015), NLG from structured data (Wen et al., 2015), dialog response generation (Vinyals and Le, 2015), among others). Overall, the workshop was held with two goals.

First, it aimed to synthesize the current state of knowledge in neural machine translation and generation: this year we continued to encourage submissions that not only advance the state of the art through algorithmic advances, but also analyze and understand the current state of the art, pointing to future research directions. Towards this goal, we received a number of high-quality research contributions on both workshop topics, as summarized in Section 2.

Second, the workshop aimed to expand the research horizons in NMT: we continued to organize the Efficient NMT task which encouraged participants to develop not only accurate but computationally efficient systems. In addition, we organized a new shared task on "Document-level Generation and Translation", which aims to push forward document-level generation technology and contrast the methods for different types of inputs. The results of the shared task are summarized in Sections 3 and 4.

2 Summary of Research Contributions

We published a call for long papers, extended abstracts for preliminary work, and cross-submissions of papers submitted to other venues. The goal was to encourage discussion and interaction with researchers from related areas.

We received a total of 68 submissions, from which we accepted 36. There were three cross-submissions, seven long abstracts and 26 full papers. There were also seven system submission papers. All research papers were reviewed twice through a double blind review process, and avoiding conflicts of interest.

There were 22 papers with an application to generation of some kind, and 14 for translation which is a switch from previous workshops where the focus was on machine translation. The caliber of the publications was very high and the number has more than doubled from last year (16 accepted papers from 25 submissions).

Proceedings of the 3rd Workshop on Neural Generation and Translation (WNGT 2019), pages 1–14
Hong Kong, China, November 4, 2019.
www.aclweb.org/anthology/D19-56%2d

3 Shared Task: Document-level Generation and Translation

The first shared task at the workshop focused on document-level generation and translation. Many recent attempts at NLG have focused on sentence-level generation (Lebret et al., 2016; Gardent et al., 2017). However, real world language generation applications tend to involve generation of much larger amount of text such as dialogues or multi-sentence summaries. The inputs to NLG systems also vary from structured data such as tables (Lebret et al., 2016) or graphs (Wang et al., 2018), to textual data (Nallapati et al., 2016). Because of such difference in data and domain, comparison between different methods has been non-trivial. This task aims to (1) push forward such document-level generation technology by providing a testbed, and (2) examine the differences between generation based on different types of inputs including both structured data and translations in another language.

In particular, we provided the following 6 tracks which focus on different input/output requirements:

- **NLG (Data → En, Data → De):** Generate document summaries in a target language given only structured data.
- **MT (De ↔ En):** Translate documents in the source language to the target language.
- **MT+NLG (Data+En → De, Data+De → En):** Generate document summaries given the structured data and the summaries in another language.

3.1 Evaluation Measures

We employ standard evaluation metrics for data-to-text NLG and MT along two axes:

Textual Accuracy Measures: We used BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004) as measures for texutal accuracy compared to reference summaries.

Content Accuracy Measures: We evaluate the fidelity of the generated content to the input data using relation generation (RG), content selection (CS), and content ordering (CO) metrics (Wiseman et al., 2017).

	Train	Valid	Test
# documents	242	240	241
Avg. # tokens (En)	323	328	329
Avg. # tokens (De)	320	324	325
Vocabulary size (En)	4163	-	-
Vocabulary size (De)	5425	-	-

Table 1: Data statistics of RotoWire English-German Dataset.

The content accuracy measures were calculated using information extraction models trained on respective target languages. We followed (Wiseman et al., 2017) and ensembled 6 information extraction models (3 CNN-based, 3 LSTM-based) with different random seeds for each language.

3.2 Data

Due to the lack of a document-level parallel corpus which provides structured data for each instance, we took an approach of translating an existing NLG dataset. Specifically, we used a subset of the RotoWire dataset (Wiseman et al., 2017) and obtained professional German translations, which are sentence-aligned to the original English articles. The obtained parallel dataset is called the RotoWire English-German dataset, and consists of box score tables, an English article, and its German translation for each instance. Table 1 shows the statistics of the obtained dataset. We used the test split from this dataset to calculate the evaluation measures for all the tracks.

We further allowed the following additional resources for each track:

- NLG: RotoWire, Monolingual
- MT: WMT19, Monolingual
- MT+NLG: RotoWire, WMT19, Monolingual

RotoWire refers to the RotoWire dataset (Wiseman et al., 2017) (train/valid), WMT19 refers to the set of parallel corpora allowable by the WMT 2019 English-German task, and Monolingual refers to monolingual data allowable by the same WMT 2019 task, pre-trained embeddings (*e.g.*, GloVe (Pennington et al., 2014)), pre-trained contextualized embeddings (*e.g.*, BERT (Devlin et al., 2019)), pre-trained language models (*e.g.*, GPT-2 (Radford et al., 2019)).

Systems which follow these resource constraints are marked constrained, otherwise unconstrained. Results are indicated by the initials (C/U).

3.3 Baseline Systems

Considering the difference in inputs for MT and NLG tracks, we prepared two baselines for respective tracks.

FairSeq-19 FairSeq (Ng et al., 2019) was used for MT and MT+NLG tracks for both directions of translations. We used the published WMT'19 single model and did not tune on in-domain data.

NCP+CC: A two-stage model from (Puduppully et al., 2019) was used for NLG tracks. We utilized the pretrained English model trained on RotoWire dataset for English article generation, while the German model was trained on RotoWire English-German dataset.

3.4 Submitted Systems

Four teams, Team EdiNLG, Team FIT-Monash, Team Microsoft, Team Naver Labs Europe, and Team SYSTRAN-AI participated in the shared task. We note the common trends across many teams and discuss the systems of individual teams below. On MT tracks, all the teams have adopted a variant of Transformer (Vaswani et al., 2017) as a sequence transduction model and trained on corpora with different data-augmentation methods. Trained systems were then fine-tuned on in-domain data including our RotoWire English-German dataset. The focus of data augmentation was two-fold: 1) acquiring in-domain data and 2) utilizing document boundaries from existing corpora. Most teams applied back-translation on various sources including NewsCrawl and the original RotoWire dataset for this purpose.

NLG tracks exhibited a similar trend for the sequence model selection, except for Team EdiNLG who employed LSTM.

3.4.1 Team EdiNLG

Team EdiNLG built their NLG system upon (Puduppully et al., 2019) by extending it to further allow copying from the table in addition to generating from vocabulary and the content plan. Additionally, they included features indicating the win/loss team records and team rank in terms of points for each player. They trained the NLG model for both languages together, using a shared BPE vocabulary obtained from target game summaries and by prefixing the target text with the target language indicator.

For MT and MT+NLG tracks, they mined the in-domain data by extracting basketball-related texts from *Newscrawl* when one of the following conditions are met: 1) player names from the RotoWire English-German training set appear, 2) two NBA team names appear in the same document, or 3) "*NBA*" appears in titles. This resulted in 4.3 and 1.1 million monolingual sentences for English and German, respectively. The obtained sentences were then back-translated and added to the training corpora. They submitted their system **EdiNLG** in all six tracks.

3.4.2 Team FIT-Monash

Team FIT-Monash built a document-level NMT system (Maruf et al., 2019) and participated in MT tracks. The document-level model was initialized with a pre-trained sentence-level NMT model on news domain parallel corpora. Two strategies for composing document-level context were proposed: flat and hierarchical attention. Flat attention was applied on all the sentences, while hierarchical attention was computed at sentence and word-level in a hierarchical manner. Sparse attention was applied at sentence-level in order to identify key sentences that are important for translating the current sentence.

To train a document-level model, the team focused on corpora that have document boundaries, including News Commentary, Rapid, and the RotoWire dataset. Notably, greedy decoding was employed due to computational cost. The submitted system is an ensemble of three runs indicated as **FIT-Monash**.

3.4.3 Team Microsoft

Team Microsoft (MS) developed a Transformer-based NLG system which consists of two sequence-to-sequence models. The two step method was inspired by the approach from (Puduppully et al., 2019), where the first model is a recurrent pointer network that selects encoded records, and the second model takes the selected content representation as input and generates summaries. The proposed model (**MS-End-to-End**) learned both models at the same time with a combined loss function. Additionally, they have investigated the use of pre-trained language models

for NLG track. Specifically, they fine-tuned GPT-2 (Radford et al., 2019) on concatenated pairs of (template, target) summaries, while constructing templates following (Wiseman et al., 2017). The two sequences are concatenated around a special token which indicates "*rewrite*". At decoding time, they adopted nucleus sampling (Holtzman et al., 2019) to enhance the generation quality. Different thresholds for nucleus sampling were investigated, and two systems with different thresholds were submitted: **MS-GPT-50** and **MS-GPT-90**, where the numbers refer to Top-p thresholds.

The generated summaries in English using the following systems were then translated with the MT systems which is described below. Hence, this marks Team Microsoft's German NLG (Data $\rightarrow$ De) submission unconstrained, due to the usage of parallel data beyond the RotoWire English-German dataset.

As for the MT model, a pre-trained system from (Xia et al., 2019) was fine-tuned on the RotoWire English-German dataset, as well as back-translated sentences from the original RotoWire dataset for the English-to-German track. Back-translation of sentences obtained from *Newscrawl* according to the similarity to RotoWire data (Moore and Lewis, 2010) was attempted but did not lead to improvement. The resulting system is shown as **MS** on MT track reports.

3.4.4 Team Naver Labs Europe

Team Naver Labs Europe (NLE) took the approach of transferring the model from MT to NLG. They first trained a sentence-level MT model by iteratively extend the training set from the WMT19 parallel data and RotoWire English-German dataset to back-translated *Newscrawl* data. The best sentence-level model was then fine-tuned at document-level, followed by fine-tuning on the RotoWire English-German dataset (constrained **NLE**) and additionally on the back-translated original RotoWire dataset (unconstrained **NLE**).

To fully leverage the MT model, input record values prefixed with special tokens for record types were sequentially fed in a specific order. Combined with the target summary, the pair of record representations and the target summaries formed data for a sequence-to-sequence model. They fine-tuned their document-level MT model on these NLG data which included the original RotoWire and RotoWire English-German dataset.

The team tackled MT+NLG tracks by concatenating source language documents and the sequence of records as inputs. To encourage the model to use record information more, they randomly masked certain portion of tokens in the source language documents.

3.4.5 Team SYSTRAN-AI

Team SYSTRAN-AI developed their NLG system based on the Transformer (Vaswani et al., 2017). The model takes as input each record from the box score featurized into embeddings and decode the summary. In addition, they introduced a content selection objective where the model learns to predict whether or not each record is used in the summary, comprising a sequence of binary classfication decision.

Furthermore, they performed data augmentation by synthesizing records whose numeric values were randomly changed in a way that does not change the win / loss relation and remains within a sane range. The synthesized records were used to generate a summary to obtain new (record, summary) pairs and were included added the training data. To bias the model toward generating more records, they further fine-tuned their model on a subset of training examples which contain $N(= 16)$ records in the summary. The submitted systems are **SYSTRAN-AI** and **SYSTRAN-AI-Detok**, which differ in tokenization.

3.5 Results

We show the results for each track in Table 2 through 7. In the NLG and MT+NLG tasks, we report BLEU, ROUGE (F1) for textual accuracy, RG (P), CS(P, R), and CO (DLD) for content accuracy. While for MT tasks, we only report BLEU. We summarize the shared task results for each track below.

In NLG (En) track, all the participants encouragingly submitted systems outperforming a strong baseline by (Puduppully et al., 2019). We observed an apparent difference between the constrained and unconstrained settings. Team NLE's approach showed that pre-training of the document-level generation model on news corpora is effective even if the source input differs (German text vs linearized records). Among constrained systems, it is worth noting that all the systems but Team EdiNLG used the Transformer, but the result did not show noticeable improvements compared to EdiNLG. It was also shown that the

generation using pre-trained language models is sensitive to how the sampling is performed; the results of MS-GPT-90 and MS-GPT-50 differ only in the nucleus sampling hyperparameter, which led to significant differences in every evaluation measure.

The NLG (De) track imposed a greater challenge compared to its English counterpart due to the lack of training data. The scores has generally dropped compared to NLG (En) results. To alleviate the lack of German data, most teams developed systems under unconstrained setting by utilizing MT resources and models. Notably, Team NLE's has achieved similar performance to the constrained system results on NLG (En). However, Team EdiNLG achieved similar performance under the constrained setting by fully leveraging the original RotoWire using the sharing of vocabulary.

In MT tracks, we see the same trend that the system under unconstrained setting (NLE) outperformed all the systems under the constrained setting. The improvement observed in the unconstrained setting came from fine-tuning on the back-translated original RotoWire dataset, which offers purely in-domain parallel documents.

While the results are not directly comparable due to different hyperparameters used in systems, fine-tuning on in-domain parallel sentences was shown effective (FairSeq-19 vs others). When incorporating document-level data, it was shown that document-level models (NLE, FIT-Monash, MS) perform better than sentence-level models (EdiNLG, FairSeq-19), even if a sentence-level model is trained on document-aware corpora.

For MT+NLG tracks, interestingly, no teams found the input structured data useful, thus applying MT models for MT+NLG tracks. Compared to the baseline (FairSeq-19), fine-tuning on in-domain data resulted in better performance overall as seen in the results of Team MS and NLE. The key difference between Team MS and NLE is the existence of document-level fine-tuning, where Team NLE outperformed in terms of textual accuracy (BLEU and ROUGE) overall, in both target languages.

4 Shared Task: Efficient NMT

The second shared task at the workshop focused on efficient neural machine translation. Many MT shared tasks, such as the ones run by the Conference on Machine Translation (Bojar et al., 2017), aim to improve the state of the art for MT with respect to accuracy: finding the most accurate MT system regardless of computational cost. However, in production settings, the efficiency of the implementation is also extremely important. The efficiency shared task for WNGT (inspired by the "small NMT" task at the Workshop on Asian Translation (Nakazawa et al., 2017)) was focused on creating systems for NMT that are not only accurate, but also efficient. Efficiency can include a number of concepts, including memory efficiency and computational efficiency. This task concerns itself with both, and we cover the detail of the evaluation below.

4.1 Evaluation Measures

We used metrics to measure several different aspects connected to how good the system is. These were measured for systems that were run on CPU, and also systems that were run on GPU.

Accuracy Measures: As a measure of translation accuracy, we used BLEU (Papineni et al., 2002) and NIST (Doddington, 2002) scores.

Computational Efficiency Measures: We measured the amount of time it takes to translate the entirety of the test set on CPU or GPU. Time for loading models was measured by having the model translate an empty file, then subtracting this from the total time to translate the test set file.

Memory Efficiency Measures: We measured: (1) the size on disk of the model, (2) the number of parameters in the model, and (3) the peak consumption of the host memory and GPU memory.

These metrics were measured by having participants submit a container for the virtualization environment Docker[1], then measuring from outside the container the usage of computation time and memory. All evaluations were performed on dedicated instances on Amazon Web Services[2], specifically of type `m5.large` for CPU evaluation, and `p3.2xlarge` (with a NVIDIA Tesla V100 GPU).

[1] https://www.docker.com/
[2] https://aws.amazon.com/

System	BLEU	R-1	R-2	R-L	RG	CS (P/R)		CO	Type
EdiNLG	17.01	44.87	18.53	25.38	91.41	30.91	64.13	21.72	C
MS-GPT-90	13.03	45.25	15.17	21.34	88.70	32.84	50.58	17.36	C
MS-GPT-50	15.17	45.34	16.64	22.93	94.35	33.91	53.82	19.30	C
MS-End-to-End	15.03	43.84	17.49	23.86	93.38	32.40	58.02	18.54	C
NLE	20.52	49.38	22.36	27.29	94.08	41.13	54.20	25.64	U
SYSTRAN-AI	17.59	47.76	20.18	25.60	83.22	31.74	44.90	20.73	C
SYSTRAN-AI-Detok	18.32	47.80	20.19	25.61	84.16	34.88	43.29	22.72	C
NCP+CC	15.80	44.83	17.07	23.46	88.59	30.47	55.38	18.31	C

Table 2: Results on the NLG: (Data → **En**) track of DGT task.

System	BLEU	R-1	R-2	R-L	RG	CS (P/R)		CO	Type
EdiNLG	10.95	34.10	12.81	19.70	70.23	23.40	41.83	16.06	C
MS-GPT-90	10.43	41.35	12.59	18.43	75.05	31.23	41.32	16.32	U
MS-GPT-50	11.84	41.51	13.65	19.68	82.79	34.81	42.51	17.12	U
MS-End-to-End	11.66	40.02	14.36	20.67	80.30	28.33	49.13	16.54	U
NLE	16.13	44.27	17.50	23.09	79.47	29.40	54.31	20.62	U
NCP+CC	7.29	29.56	7.98	16.06	49.69	21.61	26.14	11.84	C

Table 3: Results on the NLG: (Data → **De**) track of DGT task.

System	BLEU	Type
FIT-Monash	47.39	C
EdiNLG	41.15	C
MS	57.99	C
NLE	62.16	U
NLE	58.22	C
FairSeq-19	42.91	C

Table 4: DGT results on the MT track (De → En).

System	BLEU	Type
FIT-Monash	41.46	C
EdiNLG	36.85	C
MS	47.90	C
NLE	48.02	C
FairSeq-19	36.26	C

Table 5: DGT results on the MT track (En → De)

4.2 Data

The data used was from the WMT 2014 English-German task (Bojar et al., 2014), using the pre-processed corpus provided by the Stanford NLP Group[3]. Use of other data was prohibited.

4.3 Baseline Systems

Two baseline systems were prepared:

Echo: Just send the input back to the output.

Base: A baseline system using attentional LSTM-based encoder-decoders with attention (Bahdanau et al., 2015).

4.4 Submitted Systems

Two teams, Team Marian and Team Notre Dame submitted to the shared task, and we will summarize each below.

4.4.1 Team Marian

Team Marian's submission (Kim et al., 2019) was based on their submission to the shared task the previous year, consisting of Transformer models optimized in a number of ways (Junczys-Dowmunt et al., 2018). This year, they made

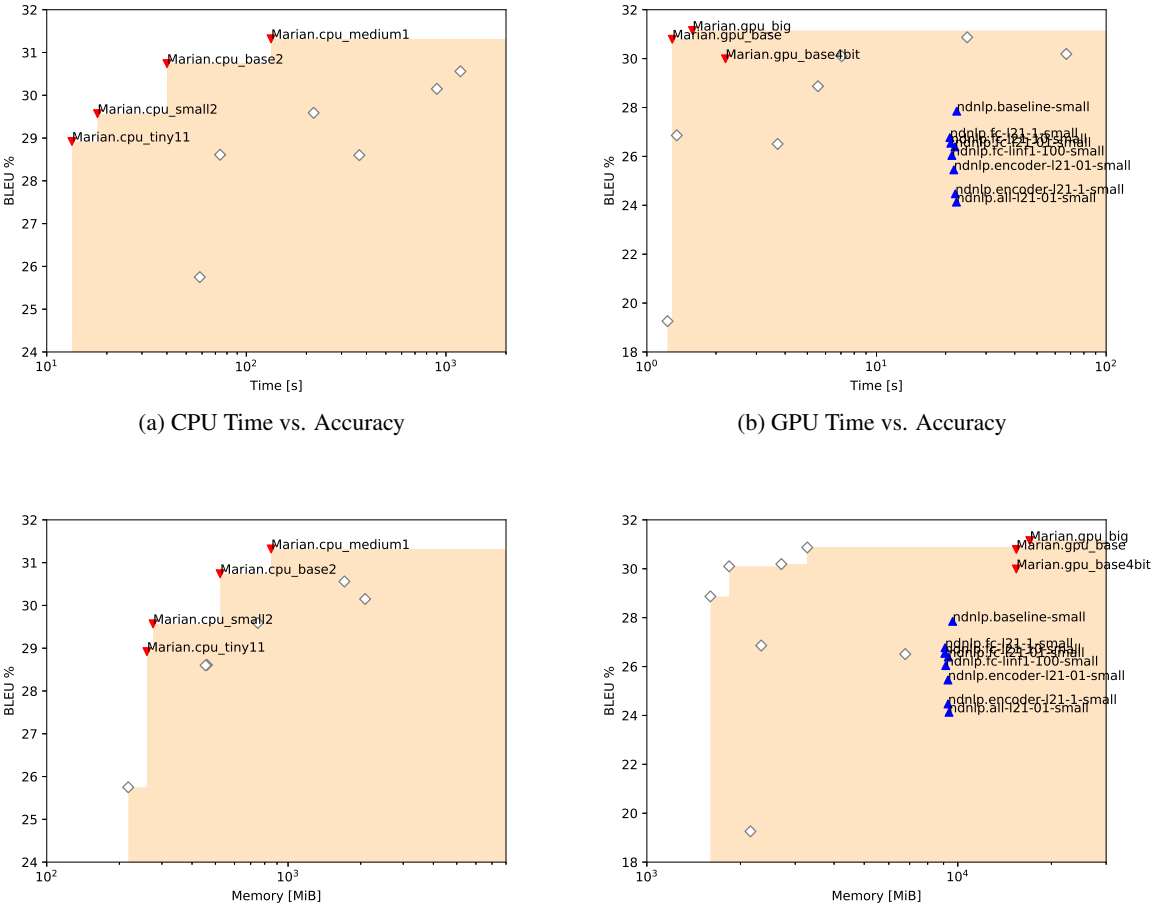
[3] https://nlp.stanford.edu/projects/nmt/

System	BLEU	R-1	R-2	R-L	RG	CS (P/R)		CO	Type
EdiNLG	36.85	69.66	41.47	57.25	81.01	77.32	78.49	62.21	C
MS	47.90	75.95	51.75	65.61	80.98	76.88	84.57	67.84	C
NLE	48.24	75.89	51.80	65.90	80.65	75.10	88.72	69.17	C
FairSeq-19	36.26	68.22	40.31	56.38	81.64	77.67	75.82	60.83	C

Table 6: Results on the MT+NLG: (Data+En → **De**) track of DGT task.

System	BLEU	R-1	R-2	R-L	RG	CS (P/R)		CO	Type
EdiNLG	41.15	76.57	50.97	66.62	91.40	78.99	63.04	51.73	C
MS	57.99	83.03	63.03	75.44	95.77	92.49	91.62	84.70	C
NLE	62.24	84.38	66.11	77.17	95.63	91.71	92.69	85.05	C
FairSeq-19	42.91	77.57	52.66	68.66	93.53	83.33	84.22	70.47	C

Table 7: Results on the MT+NLG: (Data+De → **En**) track of DGT task.



(a) CPU Time vs. Accuracy

(b) GPU Time vs. Accuracy

(c) CPU Memory vs. Accuracy

(d) GPU Memory vs. Accuracy

Figure 1: Time and memory vs. accuracy measured by BLEU on the newstest2015 set, calculated on both CPU and GPU. White diamonds (◇) represent the results in the previous campaign. Orange areas show regions dominated by some Pareto frontier systems.

a number of improvements. Improvements were made to teacher-student training by (1) creating more data for teacher-student training using backward, then forward translation, (2) using multiple teachers to generate better distilled data for training student models. In addition, there were modeling improvements made by (1) replacing simple averaging in the attention layer with an efficiently calculable "simple recurrent unit," (2) parameter tying between decoder layers, which reduces memory usage and improves cache locality on the CPU. Finally, a number of CPU-specific optimizations were performed, most notably including 8-bit matrix multiplication along with a flexible quantization scheme.

4.4.2 Team Notre Dame

Team Notre Dame's submission (Murray et al., 2019) focused mainly on memory efficiency. They did so by performing "Auto-sizing" of the transformer network, applying block-sparse regularization to remove columns and rows from the parameter matrices.

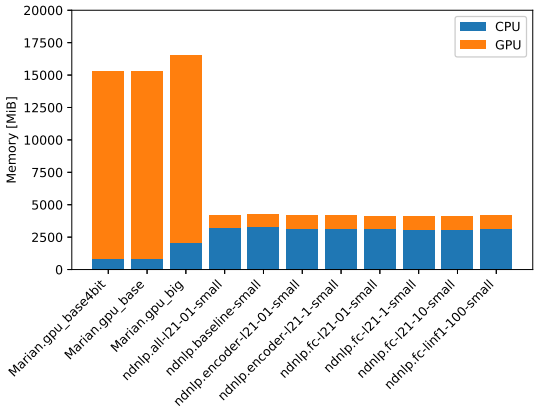
4.5 Results

A brief summary of the results of the shared task (for newstest2015) can be found in Figure 1, while full results tables for all of the systems can be found in Appendix A. From this figure we can glean a number of observations.

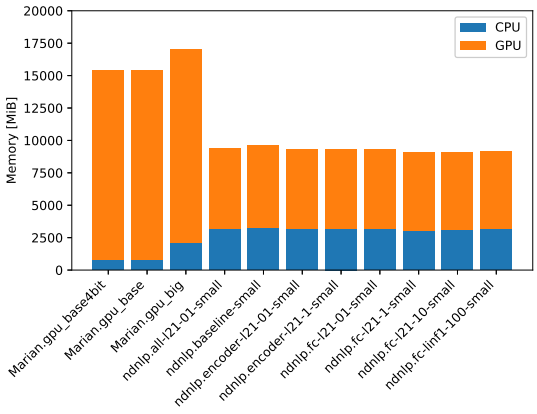
For the CPU systems, all submissions from the Marian team clearly push the Pareto frontier in terms of both time and memory. In addition, the Marian systems also demonstrated a good trade-off between time/memory and accuracy.

For the GPU systems, all systems from the Marian team also outperformed other systems in terms of the speed-accuracy trade-off. However, the Marian systems had larger memory consumption than both Notre Dame systems, which specifically optimized for memory efficiency, and all previous systems. Interestingly, each GPU system by the Marian team shares almost the same amount of GPU memory as shown in Table 12 and Figure 2(b). This may indicate that the internal framework of the Marian system tries to reserve enough amount of the GPU memory first, then use the acquired memory as needed by the translation processes.

On the other hand, we can see that the Notre Dame systems occupy only a minimal amount of GPU memory, as the systems use much smaller



(a) empty



(b) newstest2015

Figure 2: Memory consumption for each GPU system.

amounts on the empty set (Figure 2(a)). These different approaches to constant or variable size memory consumption may be based on different underlying perspectives of "memory efficiency," and it may be difficult to determine which policy is better without knowing the actual environment in which a system will be used.

5 Conclusion

This paper summarized the results of the Third Workshop on Neural Generation and Translation, where we saw a number of research advances. Particularly, this year introduced a new document generation and translation task, that tested the efficacy of systems for both the purposes of translation and generation in a single testbed.

Acknowledgments

We thank Apple and Google for their monetary support of student travel awards for the workshop, and AWS for its gift of AWS credits (to Graham Neubig) that helped support the evaluation.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. ICLR*.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, et al. 2017. Findings of the 2017 conference on machine translation (wmt17). In *Proc. WMT*, pages 169–214.

Ondrej Bojar et al. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proc. WMT*, pages 12–58.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proc. HLT*, pages 138–145.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG Challenge: Generating Text from RDF Data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The Curious Case of Neural Text Degeneration. *arXiv:1904.09751 [cs]*.

Marcin Junczys-Dowmunt, Kenneth Heafield, Hieu Hoang, Roman Grundkiewicz, and Anthony Aue. 2018. Marian: Cost-effective high-quality neural machine translation in C++. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 129–135, Melbourne, Australia. Association for Computational Linguistics.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proc. EMNLP*, pages 1700–1709.

Young Jin Kim, Marcin Junczys-Dowmunt, and Hany Hassan. 2019. From research to production and back: Fast and accurate neural machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural Text Generation from Structured Data with Application to the Biography Domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213, Austin, Texas. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Sameen Maruf, André F. T. Martins, and Gholamreza Haffari. 2019. Selective Attention for Context-aware Neural Machine Translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3092–3102, Minneapolis, Minnesota. Association for Computational Linguistics.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proc. ACL*, pages 220–224.

Kenton Murray, Brian DuSell, and David Chiang. 2019. Auto-sizing the transformer network: Shrinking parameters for the wngt 2019 efficiency task. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*.

Toshiaki Nakazawa, Shohei Higashiyama, Chenchen Ding, Hideya Mino, Isao Goto, Hideto Kazawa, Yusuke Oda, Graham Neubig, and Sadao Kurohashi. 2017. Overview of the 4th workshop on asian translation. In *Proc. WAT*, pages 1–54, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook FAIR's WMT19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 314–319, Florence, Italy. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. ACL*, pages 311–318.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-text generation with content selection and planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6908–6915.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proc. EMNLP*, pages 379–389.

Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. 2014. Sequence to sequence learning with neural networks. In *Proc. NIPS*, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. NIPS*.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869*.

Qingyun Wang, Xiaoman Pan, Lifu Huang, Boliang Zhang, Zhiying Jiang, Heng Ji, and Kevin Knight. 2018. Describing a Knowledge Base. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 10–21, Tilburg University, The Netherlands. Association for Computational Linguistics.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *Proc. EMNLP*, pages 1711–1721.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in Data-to-Document Generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263.

Yingce Xia, Xu Tan, Fei Tian, Fei Gao, Di He, Weicong Chen, Yang Fan, Linyuan Gong, Yichong Leng, Renqian Luo, Yiren Wang, Lijun Wu, Jinhua Zhu, Tao Qin, and Tie-Yan Liu. 2019. Microsoft Research Asia's Systems for WMT19. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 424–433, Florence, Italy. Association for Computational Linguistics. WMT.

A Full Shared Task Results

For completeness, in this section we add tables of the full shared task results. These include the full size of the image file for the translation system (Table 8), the comparison between compute time and evaluation scores on CPU (Table 9) and GPU (Table 10), and the comparison between memory and evaluation scores on CPU (Table 11) and GPU (Table 12).

Table 8: Image file sizes of submitted systems.

Team	System	Size [MiB]
Marian	cpu_base2	135.10
	cpu_medium1	230.22
	cpu_small2	93.00
	cpu_tiny11	91.61
	gpu_base4bit	322.68
	gpu_base	452.59
	gpu_big	773.27
Notre Dame	all-l21-01-small	2816.63
	baseline-small	2845.04
	encoder-l21-01-small	2813.67
	encoder-l21-1-small	2779.60
	fc-l21-01-small	2798.94
	fc-l21-1-small	2755.80
	fc-l21-10-small	2755.76
	fc-linf1-100-small	2759.10

Table 9: Time consumption and MT evaluation metrics (CPU systems).

Dataset	Team	System	Time Consumption [s]		BLEU %	NIST
			Total	Diff		
Empty	Marian	cpu_base2	4.97	—	—	—
		cpu_medium1	6.03	—	—	—
		cpu_small2	4.67	—	—	—
		cpu_tiny11	4.71	—	—	—
newstest2014	Marian	cpu_base2	57.52	52.55	28.04	7.458
		cpu_medium1	181.81	175.78	28.58	7.539
		cpu_small2	28.32	23.64	26.97	7.288
		cpu_tiny11	22.26	17.56	26.38	7.178
newstest2015	Marian	cpu_base2	45.01	40.04	30.74	7.607
		cpu_medium1	139.06	133.03	31.32	7.678
		cpu_small2	22.64	17.97	29.57	7.437
		cpu_tiny11	18.08	13.37	28.92	7.320

Table 10: Time consumption and MT evaluation metrics (GPU systems).

Dataset	Team	System	Time Consumption [s] Total	Diff	BLEU %	NIST
Empty	Marian	gpu_base4bit	9.77	—	—	—
		gpu_base	2.69	—	—	—
		gpu_big	5.22	—	—	—
	Notre Dame	all-l21-01-small	7.92	—	—	—
		baseline-small	8.13	—	—	—
		encoder-l21-01-small	7.96	—	—	—
		encoder-l21-1-small	7.89	—	—	—
		fc-l21-01-small	7.85	—	—	—
		fc-l21-1-small	7.56	—	—	—
		fc-l21-10-small	7.57	—	—	—
		fc-linf1-100-small	7.61	—	—	—
newstest2014	Marian	gpu_base4bit	12.42	2.66	27.50	7.347
		gpu_base	4.22	1.53	28.00	7.449
		gpu_big	7.09	1.88	28.61	7.534
	Notre Dame	all-l21-01-small	36.80	28.89	21.60	6.482
		baseline-small	36.07	27.95	25.28	7.015
		encoder-l21-01-small	35.82	27.86	23.20	6.725
		encoder-l21-1-small	35.41	27.52	22.06	6.548
		fc-l21-01-small	35.76	27.91	24.07	6.869
		fc-l21-1-small	34.46	26.90	23.97	6.859
		fc-l21-10-small	34.00	26.43	23.87	6.852
		fc-linf1-100-small	34.46	26.84	23.80	6.791
newstest2015	Marian	gpu_base4bit	11.97	2.20	29.99	7.504
		gpu_base	3.98	1.29	30.79	7.595
		gpu_big	6.80	1.58	31.15	7.664
	Notre Dame	all-l21-01-small	30.21	22.29	24.13	6.670
		baseline-small	30.47	22.35	27.85	7.224
		encoder-l21-01-small	29.67	21.71	25.45	6.869
		encoder-l21-1-small	29.93	22.04	24.47	6.734
		fc-l21-01-small	29.80	21.95	26.39	7.053
		fc-l21-1-small	28.42	20.87	26.77	7.093
		fc-l21-10-small	28.63	21.06	26.54	7.051
		fc-linf1-100-small	28.91	21.29	26.04	6.971

Table 11: Peak memory consumption (CPU systems).

Dataset	Team	System	Memory [MiB]		
			Host	GPU	Both
Empty	Marian	cpu_base2	336.30	—	336.30
		cpu_medium1	850.76	—	850.76
		cpu_small2	229.53	—	229.53
		cpu_tiny11	227.73	—	227.73
newstest2014	Marian	cpu_base2	523.93	—	523.93
		cpu_medium1	850.98	—	850.98
		cpu_small2	276.30	—	276.30
		cpu_tiny11	260.86	—	260.86
newstest2015	Marian	cpu_base2	523.12	—	523.12
		cpu_medium1	850.95	—	850.95
		cpu_small2	275.76	—	275.76
		cpu_tiny11	260.09	—	260.09

Table 12: Peak memory consumption (GPU systems).

Dataset	Team	System	Memory [MiB]		
			Host	GPU	Both
Empty	Marian	gpu_base4bit	788.67	14489	15277.67
		gpu_base	791.82	14489	15280.82
		gpu_big	2077.75	14489	16566.75
	Notre Dame	all-l21-01-small	3198.89	991	4189.89
		baseline-small	3261.73	973	4234.73
		encoder-l21-01-small	3192.87	1003	4195.87
		encoder-l21-1-small	3164.45	1003	4167.45
		fc-l21-01-small	3160.32	999	4159.32
		fc-l21-1-small	3069.05	1049	4118.05
		fc-l21-10-small	3092.01	1057	4149.01
		fc-linf1-100-small	3116.35	1037	4153.35
newstest2014	Marian	gpu_base4bit	781.83	14641	15422.83
		gpu_base	793.07	14641	15434.07
		gpu_big	2078.78	14961	17039.78
	Notre Dame	all-l21-01-small	3199.95	9181	12380.95
		baseline-small	3285.20	9239	12524.20
		encoder-l21-01-small	3194.96	9169	12363.96
		encoder-l21-1-small	3164.86	9119	12283.86
		fc-l21-01-small	3160.80	9155	12315.80
		fc-l21-1-small	3070.67	9087	12157.67
		fc-l21-10-small	3068.12	9087	12155.12
		fc-linf1-100-small	3119.96	9087	12206.96
newstest2015	Marian	gpu_base4bit	783.05	14641	15424.05
		gpu_base	789.34	14641	15430.34
		gpu_big	2077.42	14961	17038.42
	Notre Dame	all-l21-01-small	3198.86	6171	9369.86
		baseline-small	3266.11	6359	9625.11
		encoder-l21-01-small	3193.13	6103	9296.13
		encoder-l21-1-small	3166.56	6135	9301.56
		fc-l21-01-small	3160.88	6159	9319.88
		fc-l21-1-small	3079.92	6017	9096.92
		fc-l21-10-small	3069.32	6015	9084.32
		fc-linf1-100-small	3130.95	6015	9145.95