Text Summarization for CNN and DailyMail News Articles

Madhuri Sharma Akshata Bhonsle

Abstract- *This document highlights the work in the development and fine-tuning of models for summarizing news articles authored by journalists at CNN and The Daily Mail. The dataset comprises 300,000 unique English-language news articles, each accompanied by corresponding highlights and identifiers. The primary objective is to condense these articles into concise one- or two-sentence summaries, thereby facilitating the extraction of essential information and mitigating information overload. We examine both extractive and abstractive summarization techniques, using Transformers. Our methodology encompasses data preprocessing, visualization, feature engineering, model training, hyperparameter tuning, and performance evaluation to ascertain the most efficacious summarization approach.*

I. GOALS AND MOTIVATION

Background: Text summarization is the process of creating a concise and coherent version of a longer text document. The key goal is to distill the essential information, ideas, or main points, while maintaining the overall meaning and context. Over the years, text summarization has become an integral part of the field Natural Language Processing (NLP).

The concept of text summarization dates to the 1950s, with initial research focusing on simple heuristics like the frequency of word occurrence. Hans Peter Luhn, a researcher at IBM, was one of the pioneers in this field, proposing that the frequency of significant words could determine the importance of sentences. Further research in the 1970s and 1980s involved inclusion of heuristic methods and statistical approaches, where techniques like term frequency – inverse document frequency (TF-IDF) were used to identify important terms and sentences. With the 1990s came the arrival of machine learning methods and algorithms for summarization, such as the Naïve Bayes classifier and Decision Trees. From the 2000s onwards, with the advent of more advanced NLP techniques and the rise of deep learning, summarization became more sophisticated. Earlier, Recurrent Neural Networks (RNNs), particularly Long Short-Term Memory networks (LSTMs), were employed for their ability to handle sequential data. In today's world, transformer-based models such as BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-Trained Transformers) are the go-to techniques for creating more coherent, context-rich summaries.

There are two main types of text summarization: extractive and abstractive. The former approach involves selecting key sentences, phrases, or passages directly from the source text and concatenating them to form a summary. It relies on identifying the most important parts of the text without generating new sentences. On the other hand, the latter approach generates new sentences that convey key information from the source text. It involves understanding the context and rephrasing content into a shorter form, often requiring the use of advanced NLP techniques.

There are many motivations behind text summarization. In today's world, vast amounts of information are generated on a daily basis. Summarization aids users in quickly grasping the main points without having to read lengthy documents. When examining its benefits from a news content curation and delivery standpoint, it enhances productivity by allowing individuals and organizations to process large volumes of text efficiently, saving time and effort. It also makes information more accessible to a wider audience, including those with time constraints or lower literacy levels. In addition, summarization aids in quicker and more informed decision-making processes. Thus, it is key to develop good quality models that help in achieving the above goals.

Objective: This project aims to develop and fine-tune robust models capable of summarizing lengthy news articles into succinct summaries, optimizing information dissemination. The details of each model will be explained in section V. In this paper, we discuss the performance of each model based on two scores: ROUGE (Recall-Oriented Understudy for Gisting Evaluation) and BLEU (Bilingual Evaluation Understudy).

Dataset: The dataset utilized in this study comprises articles from CNN and The Daily Mail, encompassing 300,000 unique entries. Each article is paired with highlights and identifiers, providing a rich resource for training summarization models. This will be described in full detail in section III.

II. RELATED WORK

- **Literature Review**: Previous research on text summarization has explored both extractive and abstractive methodologies, as well as the metrics to evaluate them, such as ROUGE and BLEU.

  The research conducted by Papineni et al.[3] in 2002 introduces BLEU, a metric for automatically evaluating the quality of machine-translated text by comparing it to one or more reference translations. Prior to BLEU, evaluating machine translation (MT) systems relied heavily on human judgements, which were costly, time-consuming, and difficult to standardize. BLEU was developed to provide a fast, inexpensive and reproducible method for evaluating MT systems. BLEU measures the precision of n-grams (contiguous sequences of n words) in the candidate translation with respect to the reference translation(s). This involves counting the number of n-gram matches between the candidate and reference translations. To avoid favoring shorter translations that might artificially achieve high precision, BLEU introduces a brevity penalty. The penalty ensures that translations are not excessively short compared to the reference. The BLEU score combines the n-gram precision scores (typically up to 4-grams) and the brevity penalty into a single metric. Despite some limitations, such as its focus on n-gram precision and potential insensitivity to linguistic nuances, BLEU remains a foundational tool in the evaluation of MT systems and has inspired the development of subsequent evaluation metrics. It has also since been adapted as a metric to evaluate the quality of text summarization systems.

  The research conducted by Kikuchi et al.[4] in 2014 introduces a novel approach to abstractive summarization using a graph-based attentional neural network. The authors propose representing the document as a graph where nodes correspond to sentences and edges represent semantic relationships between sentences. The model uses an attention mechanism to focus on the most relevant parts of the document while generating the summary, which helps in identifying and emphasizing important content. The proposed model leverages a neural network architecture, specifically designed to handle the graph-based representation and attention mechanism. The architecture enables the model to learn complex relationships and dependencies between sentences in the document. Performance was measured using ROUGE scores, a common evaluation metric for summarization tasks. The results demonstrated that the proposed model outperformed several baseline methods, including traditional extractive and abstractive summarization techniques.

  The research conducted by See et al.[5] in 2017 introduced the pointer-generator network, designed to improve the quality of abstractive summarization by combining the strengths of both types of summarization methods. The model can choose between copying words directly from the source text (pointer) and generating new words from a fixed vocabulary (generator). This allows the model to handle out-of-vocabulary words and ensures important content is accurately included in the summary. To address the issue of word repetition in generated summaries, the authors incorporate a coverage mechanism that keeps track of which parts of the source text have been covered. This helps the model to avoid generating redundant information. Performance was measured using ROUGE scores, with the pointer-generator network achieving state-of-the-art results at the time of publication. The model demonstrated a significant improvement in generating coherent and accurate summaries compared to baseline methods.

The research conducted by Vaswani et al.[6] in 2017 introduced the transformer model, a novel architecture for sequence transduction tasks (e.g., machine translation) that relies entirely on self-attention mechanisms, dispensing with recurrence and convolutions entirely. The Transformer uses self-attention to compute representations of the input and output sequences. This mechanism allows the model to weigh the importance of different words in the input sequence for each word being processed. To capture different aspects of relationships between words, the model employs multi-head attention, which runs multiple self-attention mechanisms in parallel and concatenates their outputs. Since the model does not use recurrence, it incorporates positional encoding to inject information about the position of words in the sequence, allowing the model to understand the order of words. With an architecture comprising of an encoder, a decoder and attention mechanisms, the Transformer achieved state-of-the-art results, surpassing previous models based on recurrent and convolutional architectures in both translation quality and training efficiency. The Transformer model has revolutionized the field of natural language processing, leading to the development of numerous influential models such as BERT, GPT, and T5, which are all based on the Transformer architecture. Beyond machine translation, the Transformer architecture has been applied to a wide range of tasks, including text summarization, language modeling, and image processing, demonstrating its versatility and effectiveness.

The research of Ott et al.[7] demonstrated the use of pre-trained language models for various sequence generation tasks. The authors investigate how pre-trained checkpoints can be fine-tuned on specific downstream tasks to improve performance. The authors utilize large pre-trained models, specifically focusing on the Transformer architecture, which has been pre-trained on extensive datasets. These pre-trained models are then fine-tuned on task-specific datasets. Fine-tuning involves adjusting the weights of the pre-trained model to better fit the specific characteristics and requirements of the target task. The paper emphasizes the importance of leveraging pre-trained checkpoints, as they already contain valuable linguistic knowledge and contextual understanding from the pre-training phase, which can be transferred to downstream tasks. The performance of the fine-tuned pre-trained models is compared against models trained from scratch and other state-of-the-art approaches. Fine-tuning pre-trained models leads to significant improvements in performance across multiple sequence generation tasks. The pre-trained models achieve higher BLEU scores in machine translation and better ROUGE scores in text summarization compared to baseline models. The paper has influenced subsequent research in NLP, encouraging the adoption of pre-trained models for various tasks beyond those explicitly studied in the paper. It has highlighted the advantages of building on pre-trained architectures rather than training models from scratch.

- **Existing Techniques**: Current techniques in text summarization include traditional machine learning approaches, as well as more advanced neural network architectures such as CNNs, RNNs, and Transformers. Certain extractive summarization techniques include TextRank (which is based on the PageRank algorithm and identifies the most important sentences in a text by constructing a graph of sentences and ranking them based on their similarity), Latent Semantic Analysis (LSA) which uses Singular Value Decomposition (SVD) to identify the key concepts in text and select sentences that best highlight those concepts, traditional machine learning methods and models like Convolutional Neural Networks (CNNs) and RNNs. For abstractive summarization, Sequence-to-Sequence models (Seq2Seq), attention mechanisms, pointer-generator networks, transformer models such as Pegasus, T5 (Text-to-Text Transfer Transformer), and BERTSUM (a variant of BERT especially finetuned for summarization), and reinforcement learning have been the go-to techniques.

III. DATASET DESCRIPTION

Source: The dataset is sourced from Kaggle[1], which is in turn sourced from archives at one of the sites developed by the Computer Science department in NYU[2]. The current version supports both extractive and abstractive summarization, though the original version was created for machine reading and comprehension and abstractive question answering.

Composition and features: The dataset's structure includes comprehensive articles written in English, paired with succinct highlights that serve as ground truth for summarization tasks. The average token count for the articles were found to be 781, and 56 for the highlights. This dataset is split further into three subsets: a training set, a validation set and a testing set, consisting of 287113, 13368 and 11490 entries respectively.

According to research conducted by Bordia and Bowman in 2019, the CNN-DailyMail dataset has a slightly low gender bias compared to its counterparts, but still shows evidence of gender bias in case of words such as ‘fragile’. In addition, since these articles were written by and for the people living in the United States and the United Kingdom, they will likely present country-specific perspectives and feature events relevant to those populations during the time those articles were published.

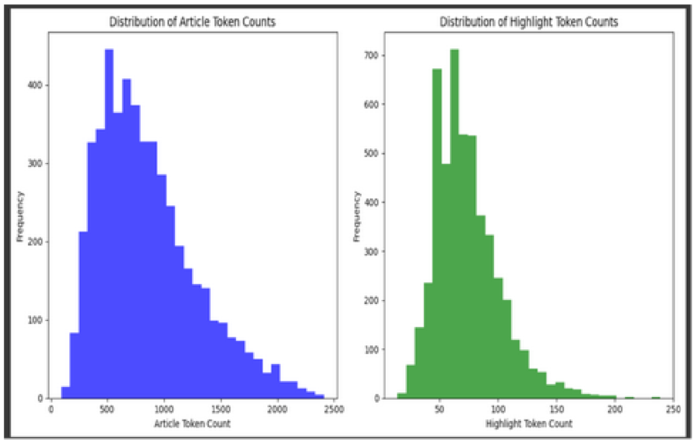
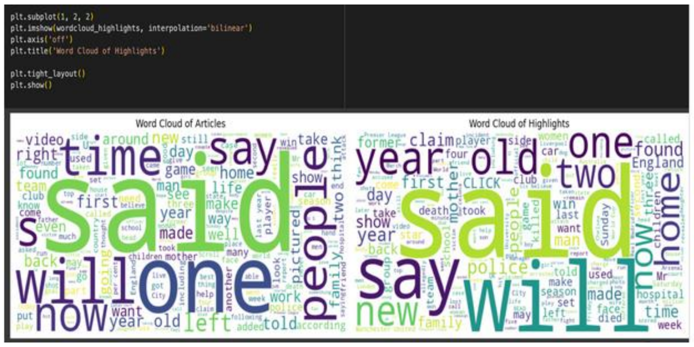


Fig 1: Token counts for articles and highlights



```
plt.subplot(1, 2, 2)
plt.imshow(wordcloud_highlights, interpolation='bilinear')
plt.axis('off')
plt.title('Word Cloud of Highlights')

plt.tight_layout()
plt.show()
```

Fig 2: The most frequent words across articles and highlights

IV. METHODOLOGY

Data Preprocessing

To ensure the data was of high quality, the data was preprocessed to handle missing data, tokenizing text, and removing stop words and punctuation.

The data was further normalized, involving converting text to lowercase, stemming, and lemmatization to standardize the dataset.

Feature Engineering

For the demonstration of the encoder-decoder architecture which was created from scratch (highlighted in section V), feature engineering was carried out by the means of word embeddings. These embeddings were extracted using BERT and applied as weights of the encoder-decoder architecture’s embedding layer.

V. MODEL IMPLEMENTATION

There were two approaches in demonstrating the summarization of the articles. The first approach involved the development of a Seq2Seq architecture to accomplish the task. After the data is cleaned and tokenized, and after feature engineering was complete, the model was created. This model consisted of an encoder and a decoder. The encoder’s embedding layer was initialized with non-trainable weights extracted from BERT. These embeddings were further fed to two LSTM layers (each having 128 units), the output of which was transferred to the decoder. The decoder consisted of an LSTM layer (also having 128 units), and a time-distributed dense layer configured with the softmax activation function. The loss function utilized was sparse categorical cross-entropy, and Adam was used as the optimizer. After the model was trained, the decoded sequences were generated.

With the second approach, various pre-trained transformers were considered, such as BART (Bidirectional and Auto-Regressive Transformer), Pegasus, T5, Marian and DistilBART. After downloading these models from the HuggingFace website, each model was fine-tuned accordingly (layers frozen, hyperparameters set, etc.) and trained on the CNN-DailyMail dataset. The optimizer used for each of these models was AdamW, a popular choice for training transformer models, alongside a learning rate scheduler that targeted reducing the learning rate if it plateaued.

VI. HYPERPARAMETER TUNING

For the first approach highlighted in section V, the various hyperparameters tuned include learning rate (ranging from 0.0001 to 0.1), batch size (ranging from 8 to 128), dropout rate (set to 0.4), recurrent dropout rate (set to 0.4) and number of training epochs (set between 50 and 100). The temperature parameter for decoding generated sequences was set to 0.1 by default.

For the second approach, the hyperparameters tuned include learning rate (ranging from 0.00005 to 0.0001), batch size (usually set to 2, ranging from 2 to 16), early stopping patience (set to 3), and the number of training epochs (set to 5 in case of BART, T5 and Pegasus, and 10 in case of Marian and DistilBART).

These hyperparameters can be summed up in the following table:

Hyperparameter	**Approach 1**	**Approach 2**
Training epochs	50-100	5-10
Learning rate	0.0001-0.1	0.00005-0.0001
Batch size	8-128	2-16
Early stopping patience	Not set	3
Dropout	0.4	Not set
Recurrent dropout	0.4	Not set
Temperature	0.1 and above	Not set

VII. RESULTS AND DISCUSSION

Performance Metrics
There were two performance metrics used for evaluating the models in case of both approaches: ROUGE and BLEU.

The primary idea behind ROUGE is to measure the overlap between the generated summary and the reference summary. There are three kinds of ROUGE scores used for evaluation: ROUGE-1 (measuring the unigram overlap between the generated and reference summaries), ROUGE-2 (measuring the bigram overlap between generated and reference summaries), and ROUGE-L (which measures the longest common subsequence between the generated and reference summaries, and emphasizes the fluency and coherence of the generated summary). A higher ROUGE score is generally considered to be indicative of a good performance by the model.

Similarly, a higher BLEU score is taken to be the sign of a well-performing model for summarization. Given the previously mentioned limitations associated with BLEU, the ROUGE score was taken to be the primary metric.

Analysis
Considering the constraints on training time and computational resources, as well as the size of the original dataset, the full dataset was not used for training. Instead, random sampling was used to create the training, validation, and test sets.

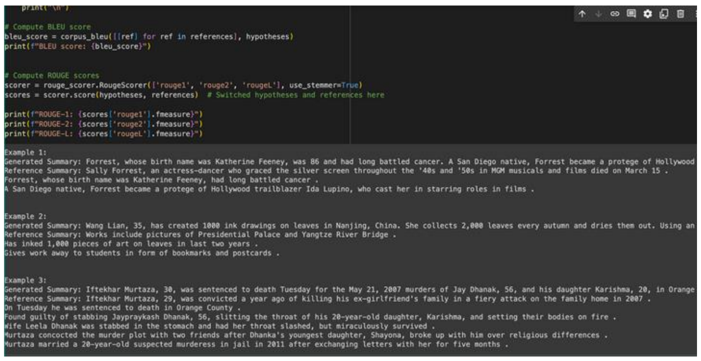
For the first approach, the number of entries in the training, validation and test sets were 6000, 2000 and 2000 respectively. The synthesized encoder-decoder architecture was trained on this dataset (with the configuration in section VI) for 100 epochs (with early stopping occurring on the 76th epoch). While the training and validation losses decreased significantly owing to the use of embeddings extracted from BERT, the summaries generated were incoherent and grammatically incorrect. In the interest of time and efficiency, the approach was not pursued further.

However, with the second approach, the number of entries in the training, validation and test datasets were 12000, 4000 and 4000 respectively. The results of fine-tuning some of the pre-trained models can be summarized from the following table:

Model	**ROUGE score (before)**	**BLEU score (before)**	**ROUGE score (after)**	**BLEU score (after)**
Marian (Helsinki-NLP/opus-mt-en-ro)	Rouge-1: 0.0705 Rouge-2: 0.0129 Rouge-L: 0.0683	0.0131	**Rouge-1: 0.3171 Rouge-2: 0.1153 Rouge-L: 0.2940**	**0.0805**
Pegasus (google/pegasus-large)	Rouge-1: 0.3231 Rouge-2: 0.1234 Rouge-L: 0.2945	0.0836	**Rouge-1: 0.3231 Rouge-2: 0.1234 Rouge-L: 0.2945**	**0.0836**
BART (facebook/bart-large)	Rouge-1: 0.2706 Rouge-2: 0.0859 Rouge-L: 0.2496	0.0592	**Rouge-1: 0.3740 Rouge-2: 0.1584 Rouge-L: 0.3499**	**0.1170**
mT5 (google/mt5-small)	Overall ROUGE score: 1.4593e-232	0.7201	**-**	**-**

It can be observed that finetuning on the pre-trained models for the smaller datasets yielded significantly better results compared to models without finetuning. This suggests that better performance could potentially be achieved through increasing the size of the dataset (providing computational resources and computing speeds are sufficient) or with further finetuning (eg: training for a longer number of epochs).

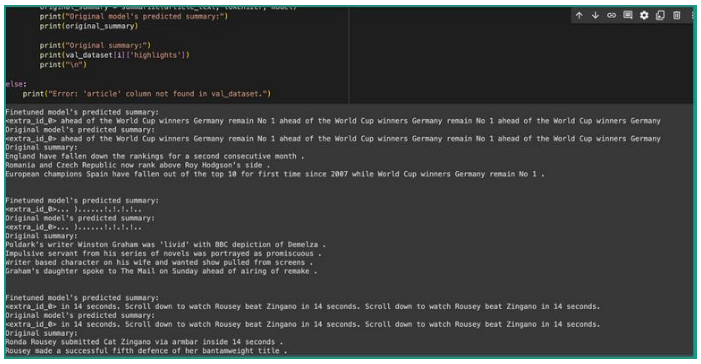
The summaries generated by these fine-tuned models are coherent and grammatically correct, as evidenced in the following figures:



```
# Compute BLEU score
bleu_score = corpus_bleu([[ref] for ref in references], hypotheses)
print(f"BLEU score: {bleu_score}")


# Compute ROUGE scores
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
scores = scorer.score(hypotheses, references)  # Switched hypotheses and references here

print(f"ROUGE-1: {scores['rouge1'].fmeasure}")
print(f"ROUGE-2: {scores['rouge2'].fmeasure}")
print(f"ROUGE-L: {scores['rougeL'].fmeasure}")
```

```
Example 1:
Generated Summary: Forrest, whose birth name was Katherine Feeney, was 86 and had long battled cancer. A San Diego native, Forrest became a protege of Hollywood
Reference Summary: Sally Forrest, an actress-dancer who graced the silver screen throughout the '40s and '50s in MGM musicals and films died on March 15 .
Forrest, whose birth name was Katherine Feeney, had long battled cancer .
A San Diego native, Forrest became a protege of Hollywood trailblazer Ida Lupino, who cast her in starring roles in films .


Example 2:
Generated Summary: Wang Lian, 35, has created 1000 ink drawings on leaves in Nanjing, China. She collects 2,000 leaves every autumn and dries them out. Using an
Reference Summary: Works include pictures of Presidential Palace and Yangtze River Bridge .
Has inked 1,000 pieces of art on leaves in last two years .
Gives work away to students in form of bookmarks and postcards .


Example 3:
Generated Summary: Iftekhar Murtaza, 30, was sentenced to death Tuesday for the May 21, 2007 murders of Jay Dhanak, 56, and his daughter Karishma, 20, in Orange
Reference Summary: Iftekhar Murtaza, 29, was convicted a year ago of killing his ex-girlfriend's family in a fiery attack on the family home in 2007 .
On Tuesday he was sentenced to death in Orange County .
Found guilty of stabbing Jaypraykash Dhanak, 56, slitting the throat of his 20-year-old daughter, Karishma, and setting their bodies on fire .
Wife Leela Dhanak was stabbed in the stomach and had her throat slashed, but miraculously survived .
Murtaza concocted the murder plot with two friends after Dhanka's youngest daughter, Shayona, broke up with him over religious differences .
Murtaza married a 20-year-old suspected murderess in jail in 2011 after exchanging letters with her for five months .
```

Fig 3: Generated summaries for BART



```
        print("Original model's predicted summary:")
        print(original_summary)

        print("Original summary:")
        print(val_dataset[i]['highlights'])
        print("\n")

else:
    print("Error: 'article' column not found in val_dataset.")
```

```
Finetuned model's predicted summary:
<extra_id_0> ahead of the World Cup winners Germany remain No 1 ahead of the World Cup winners Germany remain No 1 ahead of the World Cup winners Germany
Original model's predicted summary:
<extra_id_0> ahead of the World Cup winners Germany remain No 1 ahead of the World Cup winners Germany remain No 1 ahead of the World Cup winners Germany
Original summary:
England have fallen down the rankings for a second consecutive month .
Romania and Czech Republic now rank above Roy Hodgson's side .
European champions Spain have fallen out of the top 10 for first time since 2007 while World Cup winners Germany remain No 1 .


Finetuned model's predicted summary:
<extra_id_0>... )......!.!.!.!..
Original model's predicted summary:
<extra_id_0>... )......!.!.!.!..
Original summary:
Poldark's writer Winston Graham was 'livid' with BBC depiction of Demelza .
Impulsive servant from his series of novels was portrayed as promiscuous .
Writer based character on his wife and wanted show pulled from screens .
Graham's daughter spoke to The Mail on Sunday ahead of airing of remake .


Finetuned model's predicted summary:
<extra_id_0> in 14 seconds. Scroll down to watch Rousey beat Zingano in 14 seconds. Scroll down to watch Rousey beat Zingano in 14 seconds.
Original model's predicted summary:
<extra_id_0> in 14 seconds. Scroll down to watch Rousey beat Zingano in 14 seconds. Scroll down to watch Rousey beat Zingano in 14 seconds.
Original summary:
Ronda Rousey submitted Cat Zingano via armbar inside 14 seconds .
Rousey made a successful fifth defence of her bantamweight title .
```

Fig 4: Generated summaries for mT5

VIII. CONCLUSION

This project involved the successful development and evaluation of models for text summarization, with significant findings on the performance of pre-trained transformer models.
Future research could explore further improvements in model performance, application to other datasets, and broader domains beyond news articles.

IX. REFERENCES

1. https://www.kaggle.com/datasets/gowrishankarp/newspaper-text-summarization-cnn-dailymail/data
2. https://cs.nyu.edu/~kcho/DMQA/
3. "BLEU: a Method for Automatic Evaluation of Machine Translation", by Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu (2002)
4. “Abstractive Document Summarization with a Graph-Based Attentional Neural Model", by Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura (2014)
5. "Get To The Point: Summarization with Pointer-Generator Networks", by Abigail See, Peter J. Liu, and Christopher D. Manning (2017)
6. "Attention is All You Need", by Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin (2017)
7. "Leveraging Pre-trained Checkpoints for Sequence Generation Tasks", by Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli (2019)