# VizSeq: A Visual Analysis Toolkit for Text Generation Tasks

**Changhan Wang**[†], **Anirudh Jain***[‡], **Danlu Chen**[†] **and Jiatao Gu**[†]
[†] Facebook AI Research, [‡] Stanford University
{changhan, danluchen, jgu}@fb.com, anirudhj@stanford.edu



## Abstract

Automatic evaluation of text generation tasks (e.g. machine translation, text summarization, image captioning and video description) usually relies heavily on task-specific metrics, such as BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004). They, however, are abstract numbers and are not perfectly aligned with human assessment. This suggests inspecting detailed examples as a complement to identify system error patterns. In this paper, we present VizSeq, a visual analysis toolkit for instance-level and corpus-level system evaluation on a wide variety of text generation tasks. It supports multimodal sources and multiple text references, providing visualization in Jupyter notebook or a web app interface. It can be used locally or deployed onto public servers for centralized data hosting and benchmarking. It covers most common n-gram based metrics accelerated with multiprocessing, and also provides latest embedding-based metrics such as BERTScore (Zhang et al., 2019).

## 1 Introduction

Many natural language processing (NLP) tasks can be viewed as conditional text generation problems, where natural language texts are generated given inputs in the form of text (e.g. machine translation), image (e.g. image captioning), audio (e.g. automatic speech recognition) or video (e.g. video description). Their automatic evaluation usually relies heavily on task-specific metrics. Due to the complexity of natural language expressions, those metrics are not always perfectly aligned with human assessment. Moreover, metrics only produce abstract numbers and are limited in illustrating system error patterns. This suggests the necessity of inspecting detailed evaluation examples to get a full picture of system behaviors as well as seek improvement directions.

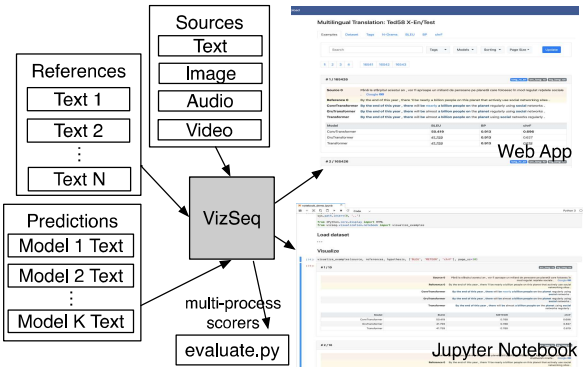


Figure 1: An overview of VizSeq. VizSeq takes multimodal sources, text references as well as model predictions as inputs, and analyzes them visually in Jupyter notebook or in a web app interface. It can also be used without visualization as a normal Python package.

A bunch of softwares have emerged to facilitate calculation of various metrics or demonstrating examples with sentence-level scores in an integrated user interface: ibleu (Madnani, 2011), MTEval [1], MT-ComparEval (Klejch et al., 2015), nlg-eval (Sharma et al., 2017), Vis-Eval Metric Viewer (Steele and Specia, 2018), compare-mt (Neubig et al., 2019), etc. Quite a few of them are collections of command-line scripts for metric calculation, which lack visualization to better present and interpret the scores. Some of them are able to generate static HTML reports to present charts and examples, but they do not allow updating visualization options interactively. MT-ComparEval is the only software we found that has an interactive user interface. It is, however, written in PHP, which unlike Python lacks a complete NLP eco-system. The number of metrics it supports is also limited and the software is no longer being actively developed. Support of multiple references is not a prevalent standard across all the softwares we investigated, and

* Work carried out during an internship at Facebook.

[1] https://github.com/odashi/mteval

| Source Type | Example Tasks |
|---|---|
| Text | machine translation, text summarization, dialog generation, grammatical error correction, open-domain question answering |
| Image | image captioning, visual question answering, optical character recognition |
| Audio | speech recognition, speech translation |
| Video | video description |
| Multimodal | multimodal machine translation |

Table 1: Example text generation tasks supported by VizSeq. The sources can be from various modalities.

| Metrics | VizSeq | compare-mt | nlg-eval | MT-Compar-Eval |
|---|---|---|---|---|
| BLEU | ✓ | ✓ | ✓ | ✓ |
| chrF | ✓ | ✓ | | |
| METEOR | ✓ | ✓ | ✓ | |
| TER | ✓ | | | ✓ |
| RIBES | ✓ | ✓ | | |
| GLEU | ✓ | | | |
| NIST | ✓ | | | |
| ROUGE | ✓ | ✓ | ✓ | |
| CIDEr | ✓ | | ✓ | |
| WER | ✓ | ✓ | | |
| LASER | ✓ | | | |
| BERTScore | ✓ | | | |

Table 2: Comparison of VizSeq and its counterparts on n-gram-based and embedding-based metric coverage.

none of them supports multiple sources or sources in non-text modalities such as image, audio and video. Almost all the metric implementations are single-processed, which cannot leverage the multiple cores in modern CPUs for speedup and better scalability.

With the above limitations identified, we want to provide a unified and scalable solution that gets rid of all those constraints and is enhanced with a user-friendly interface as well as the latest NLP technologies. In this paper, we present VizSeq, a visual analysis toolkit for a wide variety of text generation tasks, which can be used for: 1) instance-level and corpus-level system error analysis; 2) exploratory dataset analysis; 3) public data hosting and system benchmarking. It provides visualization in Jupyter notebook or a web app interface. A system overview can be found in Figure 1. We open source the software at https://github.com/facebookresearch/vizseq.

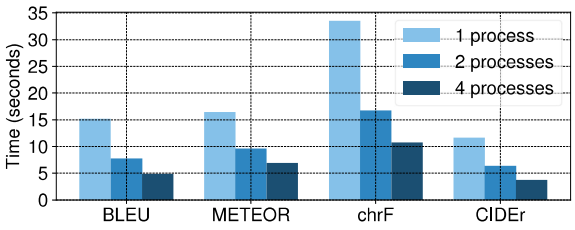


Figure 2: VizSeq implements metrics with multiprocessing speedup. Speed test is based on a 36k evaluation set for BLEU, METEOR and chrF, and a 41k one for CIDEr. CPU: Intel Core i7-7920HQ @ 3.10GHz

## 2 Main Features of VizSeq

### 2.1 Multimodal Data and Task Coverage

VizSeq has built-in support for multiple sources and references. The number of references is allowed to vary across different examples, and the sources are allowed to come from different modalities, including text, image, audio and video. This flexibility enables VizSeq to cover a wide range of text generation tasks and datasets, far beyond the scope of machine translation, which previous softwares mainly focus on. Table 1 provides a list of example tasks supported by Vizseq.

### 2.2 Metric Coverage and Scalability

Table 2 shows the comparison of VizSeq and its counterparts on metric coverage.

**N-gram-based metrics** To the extent of our knowledge, VizSeq has the best coverage of common n-gram-based metrics, including BLEU (Papineni et al., 2002), NIST (Doddington, 2002), METEOR (Banerjee and Lavie, 2005), TER (Snover et al., 2006), RIBES (Isozaki et al., 2010), chrF (Popović, 2015) and GLEU (Wu et al., 2016) for machine translation; ROUGE (Lin, 2004) for summarization and video description; CIDEr (Vedantam et al., 2015) for image captioning; and word error rate for speech recognition.

**Embedding-based metrics** N-gram-based metrics have difficulties in capturing semantic similarities since they are usually based on exact word matches. As a complement, VizSeq also integrates latest embedding-based metrics such as BERTScore (Zhang et al., 2019) and LASER (Artetxe and Schwenk, 2018). This is rarely seen in the counterparts.

**Scalability** We re-implemented all the n-gram-based metrics with multiprocessing, allowing

```python
from vizseq.scorers import
    register_scorer

@register_scorer('metric name')
def calculate_score(
    hypothesis: List[str],
    references: List[List[str]],
    n_processes: int = 2,
    verbose: bool = False
) -> Tuple[float, List[float]]:
    return corpus_score, sentence_scores
```

Figure 3: VizSeq metric API. Users can define and register their new metric by implementing this function.

users to fully utilize the power of modern multi-core CPUs. We tested our multi-process versions on large evaluation sets and observed significant speedup against original single-process ones (see Figure 2). VizSeq's embedding-based metrics are implemented using PyTorch (Paszke et al., 2017) framework and their computation is automatically parallelized on CPU or GPU by the framework.

**Versatility** VizSeq's rich metric collection is not only available in Jupyter notebook or in the web app, it can also be used in any Python scripts. A typical use case is periodic metric calculation during model training. VizSeq's implementations save time, especially when evaluation sets are large or evaluation is frequent. To allow user-defined metrics, we designed an open metric API, whose definition can be found in Figure 3.

### 2.3 User-Friendly Interface

Given the drawbacks of simple command-line interface and static HTML interface, we aim at visualized and interactive interfaces for better user experience and productivity. VizSeq provides visualization in two types of interfaces: Jupyter notebook and web app. They share the same visual analysis module (Figure 4). The web app interface additionally has a data uploading module (Figure 9) and a task/dataset browsing module (Figure 10), while the Jupyter notebook interface gets data directly from Python variables. The analysis module includes the following parts.

**Example grouping** VizSeq uses sentence tags to manage example groups (data subsets of different interest, can be overlapping). It contains both user-defined and machine-generated tags (e.g. labels for identified languages, long sentences, sentences with rare words or code-switching). Metrics will be calculated and visualized by different

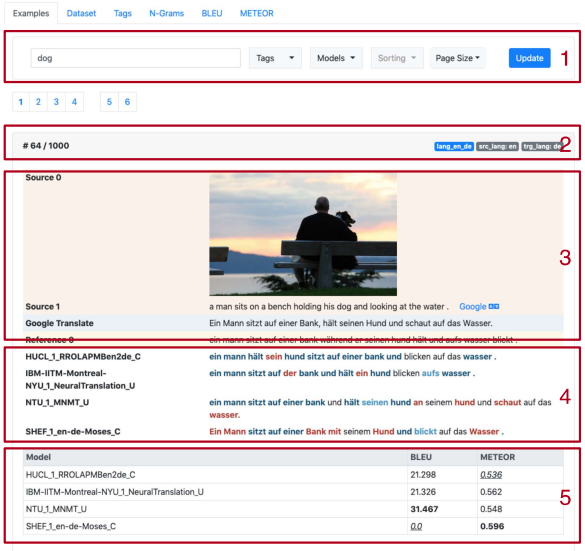


Figure 4: VizSeq example viewing. (1) keyword search box, tag and model filters, sorting and page size options; (2) left: example index, right: user-defined tags (blue) and machine-generated tags (grey); (3) multi-modal sources and Google Translate integration; (4) model predictions with highlighted matched (blue) and unmatched (red) n-grams; (5) sentence-level scores (highest ones among models in boldface, lowest ones in italics with underscore).

example groups as a complement to scores over the entire dataset.

**Example viewing** VizSeq presents examples with various sentence-level scores and visualized alignments of matched/unmatched reference n-grams in model predictions. It also has Google Translate integration to assist understanding of text sources in unfamiliar languages as well as providing a baseline translation model. Examples are listed in multiple pages (bookmarkable in web app) and can be sorted by various orders, for example, by a certain metric or source sentence lengths. Tags or n-gram keywords can be used to filter out examples of interest.

**Dataset statistics** VizSeq provides various corpus-level statistics, including: 1) counts of sentences, tokens and characters; 2) source and reference length distributions; 3) token frequency distribution; 4) list of most frequent n-grams (with links to associated examples); 5) distributions of sentence-level scores by models (Figure 5, 6 and 7). Statistics are visualized in zoomable charts with hover text hints.

**Data export** Statistics in VizSeq are one-click exportable: charts into PNG or SVG images (with

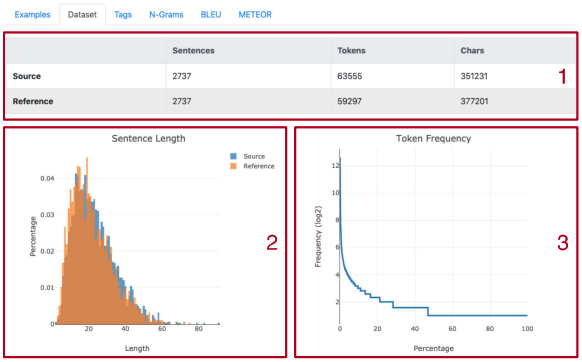


Figure 5: VizSeq dataset statistics. (1) sentence, token and character counts for source and reference sentences; (2) length distributions of source and reference sentences; (3) token frequency distribution. Plots are zoomable and exportable to SVG or PNG images.


Figure 6: VizSeq dataset statistics: most frequent n-grams (n=1,2,3,4). Each listed n-gram is clickable to show associated examples in the dataset.

users' zooming applied) and tables into CSV or LaTeX (copied to clipboard).

### 2.4 Data Management and Public Hosting

VizSeq web app interface gets new data from the data uploading module (Figure 9) or a RESTful API. Besides local deployment, the web app back-end can also be deployed onto public servers and provide a general solution for hosting public benchmarks on a wide variety of text generation tasks and datasets.

In VizSeq, data is organized by special folder structures as follows, which is easy to maintain:

```
<task>/<eval set>/source_*.{txt,zip}
<task>/<eval set>/reference_*.txt
<task>/<eval set>/tag_*.txt
<task>/<eval set>/<model>/prediction.txt
<task>/<eval set>/__cfg__.json
```

When new data comes in, scores, n-grams and machine-generated tags will be pre-computed and cached onto disk automatically. A file monitoring and versioning system (based on file hashes, sizes or modification timestamps) is employed to detect

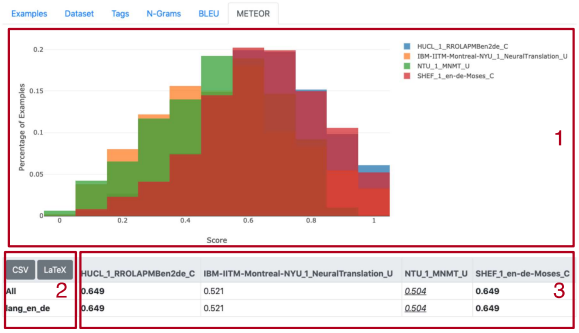


Figure 7: VizSeq corpus-level metric viewing. (1) distributions of sentence-level scores by models; (2) one-click export of tabular data to CSV and LaTeX (copied to clipboard); (3) corpus-level and group-level (by sentence tags) scores (highest ones among models in boldface, lowest ones in italics with underscore).

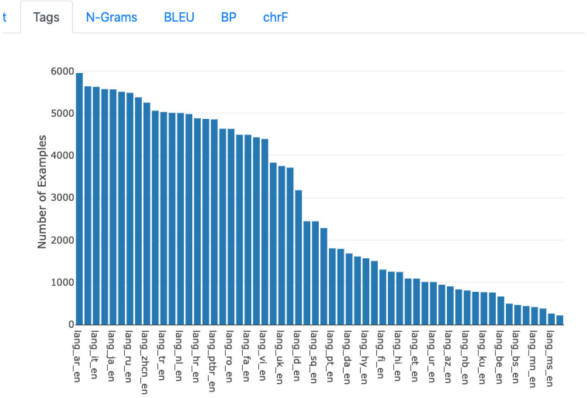


Figure 8: VizSeq sentence tag distribution view. In this example, tags are source-target language directions in a multilingual machine translation dataset.

file changes and trigger necessary updates on pre-computed results. This is important for supporting evaluation during model training where model predictions change over training epochs.

## 3 Example Use Cases of VizSeq

We validate the usability of VizSeq with multiple tasks and datasets, which are included as examples in our Github repository:

- WMT14 English-German[2]: a classic medium-size dataset for bilingual machine translation.
- Gigaword[3]: a text summarization dataset.
- COCO captioning 2015 (Lin et al., 2014): a classic image captioning dataset where VizSeq can present source images with text targets.

[2] http://www.statmt.org/wmt14/translation-task.html
[3] https://github.com/harvardnlp/sent-summary

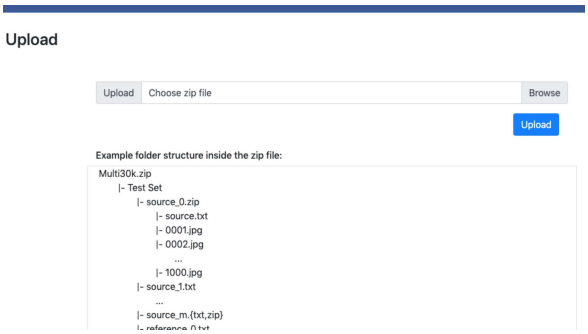


Figure 9: VizSeq data uploading. Users need to organize the files by given folder structures and pack them into a zip file for upload. VizSeq will unpack the files to the data root folder and perform integrity checks.

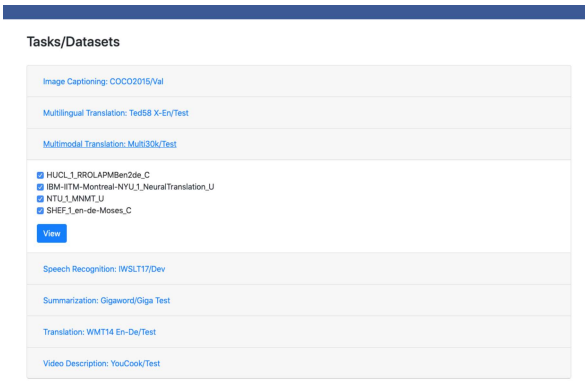


Figure 10: VizSeq task/dataset browsing. Users need to select a dataset and models of interest to proceed to the analysis module.

- WMT16 multimodal machine translation task 1[4]: English-German translation with an image the sentences describe. VizSeq can present both text and image sources, and calculate the official BLEU, METEOR and TER metrics.
- Multilingual machine translation on TED talks dataset (Ye et al., 2018): translation from 58 languages into English. VizSeq can use language directions as sentence tags to generate score breakdown by languages. The test set has as many as 165k examples, where VizSeq multi-process scorers run significantly faster than single-process ones. The integrated Google Translate can help with understanding source sentences in unfamiliar languages.
- IWSLT17 English-German speech translation[5]: VizSeq can present English audios with English transcripts and German text translations.
- YouCook (Das et al., 2013) video description: VizSeq enables inspecting generated text descriptions with presence of video contents.

[4] https://www.statmt.org/wmt16/multimodal-task.html
[5] https://sites.google.com/site/iwsltevaluation2017

## 4 Related Work

With the thrive of deep learning, task-agnostic visualization toolkits such as Tensorboard[6], visdom[7] and TensorWatch[8], have emerged in need of monitoring model statistics and debugging model training. Model interpretability is another motivation for visualization. In NLP, softwares have been developed to interpret model parameters (e.g. attention weights) and inspect prediction generation process: LM (Rong and Adar, 2016), OpenNMT visualization tool (Klein et al., 2018) and Seq2Seq (Strobelt et al., 2019). For machine translation, toolkits are made to perform metric calculation and error analysis: ibleu (Madnani, 2011), MTEval [9], MT-ComparEval (Klejch et al., 2015), nlg-eval (Sharma et al., 2017), Vis-Eval Metric Viewer (Steele and Specia, 2018) and compare-mt (Neubig et al., 2019).

## 5 Conclusion

In this paper, we present VizSeq, a visual analysis toolkit for {text, image, audio, video}-to-text generation system evaluation, dataset analysis and benchmark hosting. It is accessible as a web app or a Python package in Jupyter notebook or Python scripts. VizSeq is currently under active development and our future work includes: 1) enabling image-to-text and video-to-text alignments; 2) adding human assessment modules; 3) integration with popular text generation frameworks such as fairseq[10], opennmt[11] and tensor2tensor[12].

## Acknowledgments

We thank the anonymous reviewers for their comments. We also thank Ann Lee and Pratik Ringshia for helpful discussions on this project.

## References

Mikel Artetxe and Holger Schwenk. 2018. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *arXiv preprint arXiv:1812.10464*.

[6] https://github.com/tensorflow/tensorboard
[7] https://github.com/facebookresearch/visdom
[8] https://github.com/microsoft/tensorwatch
[9] https://github.com/odashi/mteval
[10] https://github.com/pytorch/fairseq
[11] https://github.com/OpenNMT/OpenNMT-py
[12] https://github.com/tensorflow/tensor2tensor

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Pradipto Das, Chenliang Xu, Richard F Doell, and Jason J Corso. 2013. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2634–2641.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145. Morgan Kaufmann Publishers Inc.

Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010. Automatic evaluation of translation quality for distant language pairs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 944–952. Association for Computational Linguistics.

Guillaume Klein, Yoon Kim, Yuntian Deng, Vincent Nguyen, Jean Senellart, and Alexander M. Rush. 2018. Opennmt: Neural machine translation toolkit.

Ondřej Klejch, Eleftherios Avramidis, Aljoscha Burchardt, and Martin Popel. 2015. Mt-compareval: Graphical evaluation interface for machine translation development. *The Prague Bulletin of Mathematical Linguistics*, 104(1):63–74.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Nitin Madnani. 2011. ibleu: Interactively debugging and scoring statistical machine translation systems. In *2011 IEEE Fifth International Conference on Semantic Computing*, pages 213–214. IEEE.

Graham Neubig, Zi-Yi Dou, Junjie Hu, Paul Michel, Danish Pruthi, and Xinyi Wang. 2019. compare-mt: A tool for holistic comparison of language generation systems. *arXiv preprint arXiv:1903.07926*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*.

Maja Popović. 2015. chrf: character n-gram f-score for automatic mt evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395.

Xin Rong and Eytan Adar. 2016. Visual tools for debugging neural language models. In *Proceedings of ICML Workshop on Visualization for Deep Learning*.

Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. 2017. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *CoRR*, abs/1706.09799.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200.

David Steele and Lucia Specia. 2018. Vis-eval metric viewer: A visualisation tool for inspecting and evaluating metric scores of machine translation output. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 71–75.

Hendrik Strobelt, Sebastian Gehrmann, Michael Behrisch, Adam Perer, Hanspeter Pfister, and Alexander M Rush. 2019. Seq2seq-vis: A visual debugging tool for sequence-to-sequence models. *IEEE transactions on visualization and computer graphics*, 25(1):353–363.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

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

Qi Ye, Sachan Devendra, Felix Matthieu, Padmanabhan Sarguna, and Neubig Graham. 2018. When and why are pre-trained word embeddings useful for neural machine translation. In *HLT-NAACL*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.