KPQA: A Metric for Generative Question Answering Using Word Weights

Hwanhee Lee¹, Seunghyun Yoon¹, Franck Dernoncourt²
Doo Soon Kim², Trung Bui², Joongbo Shin¹ and Kyomin Jung¹
¹Dept. of Electrical and Computer Engineering, Seoul National University, Seoul, Korea
²Adobe Research, San Jose, CA, USA
{wanted1007, mysmilesh, jbshin, kjung}@snu.ac.kr
{franck.dernoncourt, dkim, bui}@adobe.com

Abstract

For the automatic evaluation of Generative Question Answering (genQA) systems, it is essential to assess the correctness of the generated answers. However, n-gram similarity metrics, which are widely used to compare generated texts and references, are prone to misjudge fact-based assessments. Moreover, there is a lack of benchmark datasets to measure the quality of metrics in terms of the correctness. To study a better metric for genQA, we collect high-quality human judgments of correctness on two standard genQA datasets. Using our human-evaluation datasets, we show that existing metrics based on n-gram similarity do not correlate with human judgments. To alleviate this problem, we propose a new metric for evaluating the correctness of genQA. Specifically, the new metric assigns different weights on each token via keyphrase prediction, thereby judging whether a predicted answer sentence captures the key meaning of the human judge’s ground-truth. Our proposed metric shows a significantly higher correlation with human judgment than widely used existing metrics.

1 Introduction

Question answering (QA) system has received consistent attention in the natural language processing community. Recently, research on QA systems has reached the stage of generating answers, called genQA, rather than to extract the answer from the context for a given question. (Yin et al., 2015; Bauer et al., 2018; Nishida et al., 2019; Bi et al., 2019) However, a bottleneck in developing genQA models is that there is no proper automatic metrics to evaluate generated answer. (Chen et al., 2019)

In evaluating the genQA model, it is essential to consider whether the generated response correctly contains vital information to answer the question. There exist several n-gram similarity metrics such as BLEU (Papineni et al., 2002) and ROUGE-L (Lin, 2004), measuring word overlaps between generated response and ground-truth answer; however, these metrics are insufficient to evaluate the genQA system. (Yang et al., 2018; Chen et al., 2019). For instance, in the example in Fig. 1 from the AVSD (Alamri et al., 2019) dataset where widely used n-gram similarity metrics does not align with human judgements of correctness. All scores are between 0 and 1.

Figure 1: An Example from Audio Visual Scene-aware Dialog (AVSD) (Alamri et al., 2019) dataset where widely used n-gram similarity metrics do not align with human judgements of correctness. All scores are between 0 and 1.
Also, we find another limitation that there is no proper benchmark dataset to evaluate automatic evaluation metrics for assessing the correctness of generated response in genQA. To fill this gap, we firstly collect human evaluations for the correctness of generated answers obtained from state-of-the-art models on two standard genQA datasets. By designing careful instructions and filtering noisy annotations, we create high-quality datasets for evaluating the correctness in the genQA domain. With the proposed datasets, we confirm that existing metrics are poorly correlated with human judgments in the preliminary experiment.

To overcome the shortcomings of previous metrics, we develop a novel keyphrase predictor, which computes the importance weight of each word in both predicted answer and ground-truth answer when evaluating its correctness. By integrating the output from the keyphrase predictor, we propose a KPQA-metric, which assigns high weight to an important word when assessing correctness. Our KPQA-metric is computed in two steps: (1) Given a {question, generated answer, reference answer}, we compute importance weights for each question-answer pair {question, generated answer} and {question, ground-truth answer} using a pre-trained keyword prediction model; (2) By using the importance weights, we then compute a weighted similarity score by integrating it into existing metrics. Our approach can be easily integrated into most of the existing metrics, including n-gram similarity metrics and recently proposed BERTScore (Zhang et al., 2019). We evaluate the proposed method on four datasets: MS-MARCO (Nguyen et al., 2016), AVSD (Alamri et al., 2019), NarrativeQA (Kočiský et al., 2018) and SemEval (Ostermann et al., 2018). To evaluate our proposed method, we newly collect human judgments for MS-MARCO and AVSD from the hired annotators. For NarrativeQA and SemEval dataset, we use the data from (Chen et al., 2019), who also studied metrics for QA. Our experimental results show that the proposed metric has significantly higher correlations with human judgments than those of the previous metrics for all of the four datasets. Also, our importance weighting mechanism has strong interpretability since the importance weights show where to focus as visualized in Figure 3. We will release the human-annotated benchmark dataset and pre-trained models to compute KPQA-metric for the research community.

2 Preliminary: Automated Text Evaluation Metrics

We briefly review the current automated text evaluation metrics which have been used for evaluating the genQA systems.

**BLEU** is a popular evaluation metric for generated text based on n-gram precision. BLEU scores a candidate by counting the number present in the reference among the n-gram of the candidate. In general, n is varied from 1 to 4, and the scores for varying n are aggregated with a geometric mean. In this work, we look at BLEU-1 and BLEU-4, where $n = 1$ and $n = 4$, respectively.

**ROUGE** is a set of evaluation metrics used for automatic text generation such as summarization and machine translation. Typically, most studies used ROUGE-L, which is a F-measure based on the L longest common subsequence between a candidate and the reference. Unlike BLEU, ROUGE-L has the advantage of not requiring the predefined number of contiguous sequence $n$.

**METEOR** (Banerjee and Lavie, 2005) is an F1 score of a set of unigram alignments. METEOR has a unique property that it considers stemmed words, synonyms, and paraphrases, as well as the standard exact word matches.

**BERTScore** is a recently proposed text evaluation metric using pre-trained representations from BERT (Devlin et al., 2019). BERTScore firstly computes the contextual embeddings for given references and predictions independently with BERT, and then computes pairwise cosine similarity scores.

3 Collecting Human judgements

To evaluate the evaluation metrics, we collect human judgement scores for two of genQA datasets. The human scores can be used to measure the correlation between human judgements and evaluation metrics.

3.1 Datasets

Recently, Chen et al. (2019) introduced human judgements for genQA in two datasets, NarrativeQA (Kočiský et al., 2018) and SemEval 2018 Task 11 (Ostermann et al., 2018). We find that the average lengths of the answer sentence are 4.7 and 2.5 for NarrativeQA and SemEval 2018 Task, respectively, as shown in Table 1. These short answers cannot be representative of genQA, because
the answers could be long and may deliver complex meaning. To fill the gap, we collect human judgements of correctness for model predictions on two genQA datasets, MS-MARCO (Nguyen et al., 2016) and AVSD (Alamri et al., 2019). We argue that MS-MARCO and AVSD, which have longer answers than NarrativeQA and SemEval 2018 Task 11, are more suitable for studying the metrics for general form of genQA.

### 3.2 Collecting Human judgement of Answer Correctness

We first obtain the model predictions by training QA models for the target datasets and generating the answers for the test sets. In the experiments, two best performing models are employed for each dataset, UniLM (Dong et al., 2019) and MTN (Le et al., 2019) for MS-MARCO and AVSD, respectively. We further provide detailed information on these two models in Appendix B. To prepare the test set, we randomly select 300 samples from the development set of MS-MARCO and 300 samples in the test set for AVSD. The generated responses are evaluated by humans to annotate the correctness of the predictions compared to ground-truth answers. In the following section, we will describe the procedure of collecting human judgements.

**Instructions to Annotators** We hire the workers from Amazon Mechanical Turk (MTurk) to rate the correctness of the generated answers from the models we trained. We assign ten workers for each sample to get reliable data. The instructions are shown in Fig. 2. We request the workers to annotate the correctness using a 5-point Likert scale (Wikipedia, 2020), where 1 means completely wrong, and 5 means completely correct.

**Filtering Noisy Workers** Some workers did not follow the instructions, producing poor-quality judgements. To solve this problem, we filter noisy responses using z-score as in (Jung and Lease, 2011). We first compute the z-score among ten responses for each sample. Then, we consider the responses whose z-score is higher than 1 as noise and remove them up to five in the order of the z-score. As a result, all of the samples have at least five annotations after removing the noisy responses. The average number of annotators is shown in Table 2. We use the average score of the annotators for each sample as a ground-truth evaluation score to assess the quality of the evaluation metric.

**Inter-Annotator Agreement** The final dataset is further validated with Krippendorff’s alpha (Krippendorff, 2011), a statistical measure of inter-rater agreement for multiple annotators. We observe that the Krippendorff’s α is higher than 0.8 for both datasets, as shown in Table 2. These coefficient numbers indicate a “near-perfect” agreement according to one of the general guidelines (Landis and Koch, 1977) for kappa-like measures.

### 4 Proposed Metric for Evaluating genQA

To build a better metric for the genQA, we first propose a Keyphrase Predictor for Question Answering (KPQA). By considering the question, the KPQA assigns different weights to each token in the answer sentence in such a way that salient to-
kens receive a high value. We integrate the KPQA to existing metrics to make them evaluate the correctness as well.

4.1 Keyphrase Predictor for Question Answering

For genQA, we observe that each word has different importance when assessing a generated answer. As shown in Fig. 1, there exist keywords or key-phrases that are considered significant while evaluating the correctness of the answer. Also, articles such as “a” and “the” are mostly irrelevant to the correctness of the answer. To predict the importance of each word we introduce Keyphrase Predictor for Question Answering (KPQA).

As shown in Fig. 3, KPQA is a BERT-based (Devlin et al., 2019) classifier that can predict salient tokens in the answer sentence depending on the question. We regard it as a multi-class classification task where each token is a single class. To train KPQA, we first prepare extractive QA datasets such as SQuAD (Rajpurkar et al., 2016), GQA (Hudson and Manning, 2019) and FSVQA (Shin et al., 2016), which consist of {passage, question, answer-span}. These datasets are transformed into pairs of {answer-sentence, question, answer-span}. The answer-sentence is extracted from the passage so that it contains answer-span in it. The question and answer-sentence are concatenated and fed into KPQA to consider the question while classifying the salient tokens in the answer-sentence.

4.2 KPQA Metric

Since KPQA’s training process allows KPQA to find essential words in the answer sentence to a given question, we use pre-trained KPQA to get the importance weights that are useful for evaluating the correctness of generated answers in genQA. We describe how we combine these weights to existing metrics to derive the KPQA-metric.

We first compute the importance weights for a given question \( Q = (q_1, ..., q_l) \), predicted answer \( X = (x_1, ..., x_n) \) and reference answer \( \hat{X} = (\hat{x}_1, ..., \hat{x}_n) \) using pre-trained KPQA. We provide each pair \{question, generated answer\} and \{question, ground-truth answer\} to pre-trained KPQA and get the output of the softmax layer after [SEP]. We define these parts as KeyPhrase Weight, KPW as shown in Fig. 3. We note that \( \text{KPW}(Q, X) = (w_1, ..., w_m) \) is an importance weight of prediction \( X \) given question \( Q \). These weights reflect the importance of each token for evaluating the correctness of a given full-sentence answer \( X \) for the given question \( Q \). We set KPW to 0 for tokens that are stopwords, such as “a” or “the”, to ignore them. The list of the stopwords and other implementations details are in Appendix C. We then compute KPQA-metric by incorporating KPW to several existing metrics modifying the precision and recall to compute weighted similarity.

**ROUGE-L-KPQA** For instance, we derive ROUGE-L-KPQA, which is a modified version of ROUGE-L using KPW to compute weighted precision(\( P_{LCS}^{KPQA} \)), recall(\( R_{LCS}^{KPQA} \)) and F1(\( F_{LCS}^{KPQA} \)), as follows:

\[
P_{LCS}^{KPQA} = \frac{\sum_{i=1}^{n} LCS^{KPQA}(\hat{x}_i, X)}{\sum_{i=1}^{m} \text{KPW}_i^{(Q,X)}},
\]

\[
R_{LCS}^{KPQA} = \frac{\sum_{i=1}^{n} LCS^{KPQA}(\hat{x}_i, X)}{\sum_{i=1}^{m} \text{KPW}_i^{(Q,X)}},
\]

\[
F_{LCS}^{KPQA} = \frac{2 \times P_{LCS}^{KPQA} \times R_{LCS}^{KPQA}}{P_{LCS}^{KPQA} + R_{LCS}^{KPQA}}.
\]
| Metrics          | MS-MARCO | AVSD | NarrativeQA | SemEval |
|------------------|----------|------|-------------|---------|
|                 | r        | ρ    | r           | r       |
| BLEU-1           | 0.306    | 0.319| 0.467       | 0.481   |
| BLEU-4           | 0.158    | 0.205| 0.418       | 0.478   |
| ROUGE-L          | 0.306    | 0.313| 0.503       | 0.569   |
| METEOR           | 0.414    | 0.431| 0.564       | 0.734   |
| BERTScore        | 0.488    | 0.512| 0.587       | 0.785   |
| ROUGE-L-KPQA     | 0.582    | 0.616| 0.723       | 0.709   |
| BERTScore-KPQA   | 0.652    | 0.646| 0.728       | 0.705   |

Table 3: Correlation between metrics and human judgments of correctness using Pearson Correlation(r), Spearman’s Correlation(ρ). Some of the results of NarrativeQA and SemEval are from (Chen et al., 2019).

\[
F^{KPCQA}_{LCS} = \frac{(1 + \beta^2) R^{KPCQA}_{LCS} P^{KPCQA}_{LCS}}{R^{KPCQA}_{LCS} + \beta^2 P^{KPCQA}_{LCS}}, \quad (3)
\]

where LCS is the Longest Common Subsequence between a prediction and a reference. The \( LCS^{KPCQA}(\hat{x}_i, X) \) is defined as follows:

\[
LCS^{KPCQA}(\hat{x}_i, X) = \sum_{i=1}^{m} I_i \cdot KPW_i^{(Q, \hat{x})}, \quad (4)
\]

where \( I_i \) is an indicator function which is 1 if each word is in the LCS and 0 otherwise.

**BERTScore-KPQA** Similarly deriving ROUGE-L-KPQA, we compute BERTScore-KPQA using KPW. We first compute contextual embedding \( x \) for a prediction \( X \) and \( \hat{x} \) for a reference \( \hat{X} \) using the BERT model. Then, we compute weighted precision\( P^{KPCQA}_{BERT} \), recall\( R^{KPCQA}_{BERT} \) and \( F^1_{BERT} \) with contextual embedding and KPW of each token as follows:

\[
P^{KPCQA}_{BERT} = \frac{\sum_{i=1}^n KPW_i^{(Q,X)} \cdot \max_{\hat{x}_j \in \hat{x}} x_i^T \hat{x}_j}{\sum_{i=1}^n KPW_i^{(Q,X)}}, \quad (5)
\]

\[
R^{KPCQA}_{BERT} = \frac{\sum_{i=1}^n KPW_i^{(Q,\hat{x})} \cdot \max_{x_j \in x} x_i^T \hat{x}_j}{\sum_{i=1}^n KPW_i^{(Q,\hat{x})}}, \quad (6)
\]

\[
F^1_{BERT} = 2 \cdot \frac{P^{KPCQA}_{BERT} \cdot R^{KPCQA}_{BERT}}{P^{KPCQA}_{BERT} + R^{KPCQA}_{BERT}}, \quad (7)
\]

| Dataset | F1 |
|---------|----|
| SQuAD   | 57.83 |
| GQA     | 99.74 |
| FSVQA   | 99.61 |

Table 4: Performance of our keyphrase predictor in development set of each dataset.

(SQuAD (Rajpurkar et al., 2016), GQA (Hudson and Manning, 2019), FSVQA (Shin et al., 2016)) to build a general keyphrase extractor for question answering. For the SQuAD dataset, we select the sentence that include a short answer span as a full-sentence answer. For GQA dataset and FSVQA dataset, both of the datasets provide full-sentence answers and short answers. We construct the trainset for KPQA by randomly extracting 75k samples from each of the three datasets to balance the number of samples for each dataset. We then train a model for two epochs on the combined dataset. The performance of our keyword predictor in the development set, which is randomly extracted 10k samples from each of three datasets, is shown in Table 4.

**BERTScore** For BERTScore we use bert-large-uncased-whole-word-mask-finetuned-squad, (24 layers, 1024 hidden units, 16 heads) from (Wolf et al., 2019) which is a BERT model fine-tuned on QA dataset SQuAD. We observe that computing BERTScore through this BERT model shows slightly higher correlation with human judgements than the BERT model without fine tuning. We use the first layer of it after the word embedding layer to compute the embedding. We experiment among different layers and found that the first hidden layer yielded the best result.

5 Experimental Results

### 5.1 Implementation Details

**Keyphrase Predictor** We train the single keyphrase predictor with various datasets
5.2 Evaluation Methods for Metrics

To compare the performance of various existing metrics and our metric, we use the Pearson coefficient and Spearman coefficient. We compute these correlation coefficients with human judgements of correctness. We test with MS-MARCO, AVSD which we collected human judgements and also for the datasets Narrative QA and SemEval from (Chen et al., 2019).

5.3 Performance Evaluation

Table 3 shows the correlation scores for the baseline metrics and KPQA-augmented ones for multiple datasets. The correlation between human judgement and most of the existing metrics such as BLEU or ROUGE-L is very low, especially for the MS-MARCO dataset, which has longer and more abstractive answers than the other three datasets. Hence, most of the widely used n-gram similarity metrics are inadequate to evaluate the correctness of the answer for genQA especially for the datasets that have abstractive answer. We also observe a higher correlation score for our proposed KPQA-metric, BERTScore-KPQA and ROUGE-L-KPQA, compared to existing metrics including original BERTScore and ROUGE-L. We observe that this gap of correlation is especially higher for MS-MARCO dataset and we argue that our proposed metric is especially more effective in evaluating abstractive answers than existing metrics.

5.4 Comparison with IDF

The next best metric after our proposed metric is the original BERTScore, which uses contextual embeddings and adopt IDF (Inverse Document Frequency) based importance weighting. When computing the original BERTScore, we compute the IDF dictionary with reference texts and adopt importance weighting with IDF as in (Zhang et al., 2019). One of our KPQA-metrics, BERTScore-KPQA, which uses KP as importance weights instead of IDF, outperforms original BERTScore with a significant gap as shown in Table 3. By comparing BERTScore-KPQA and BERTScore, we show that our importance weighting method using KPQA is more effective than IDF for evaluating correctness. Since IDF is dependent on the frequency, it assigns a lower weight to some important words that frequently occur in the reference sentence. Hence, IDF based importance weighting might not be helpful for some cases. On the other hand, our KPW is computed utilizing questions so that it can assign the weights to words in the context of the questions in the generated answer, and this leads to better correlation with human evaluation.

5.5 Ablation Study

To validate the effectiveness of our KPW, we perform several ablation studies and present results in Table 5. For the results in the second row, we substitute KPW in BERTScore-KPQA with uniform weight whose weight for each token is set to one. By doing this, we can see the effect of our importance weighting by KPW that is conditioned on the question. We can observe that performance of KPQA-metric is higher than uniform weights. This gap is especially higher for the MS-MARCO dataset, where the average number of tokens is longer than other datasets. In the third row in Table 5, we can see the effect of stopword removal. Since our KPW already assigns lower weights to unnecessary words, the effect of setting the weight of stopwords to zero to remove them is slight and even negative for SemEval dataset. But it is usually effective to make stopwords’ weight zero so that it cannot be used completely in other three datasets.

6 Related Work

One important next steps for current QA systems are the systems that can generate long answers in natural language for given question and context. Following this interests, several generative (abstractive) QA datasets (Nguyen et al., 2016; He et al., 2017; Kočiský et al., 2018), where the answer is not necessarily in the passage, were recently released. Since the task is to generate natural language for the given question, the QA system is often trained with seq2seq (Sutskever et al., 2014) objective similarly to other natural generation tasks such as neural machine translation. Hence, researchers often use n-gram based similarity metrics such as BLEU to evaluate the genQA systems, following other natural language generation tasks.

However, most of these n-gram metrics including BLEU are originally developed for evaluating machine translation and previous works (Liu et al., 2016; Nema and Khapra, 2018; Kryscinski et al., 2019) showed that these metrics have poor correlation with human judgements in other language generation tasks such as dialog systems. Like other text generation systems, it is difficult to assess the performance through n-gram metrics for genQA.
| Metrics                  | MS-MARCO | AVSD | NarrativeQA | SemEval |
|-------------------------|----------|------|-------------|---------|
|                         | r       | ρ   | r           | ρ      | r      | ρ   |
| BERTScore-KPQA          | 0.652   | 0.646 | 0.728       | 0.705  | 0.790  | 0.770 |
| - KPW                   | 0.497   | 0.538 | 0.600       | 0.618  | 0.776  | 0.761 |
| - stopwords removal      | 0.611   | 0.618 | 0.696       | 0.676  | 0.778  | 0.767 |

Table 5: Ablation studies for our proposed metrics with human judgements.

For genQA, n-gram similarity metrics can give high scores to the generated answer that is incorrect but contains a lot of unnecessary words in the reference answer. Previous works (Yang et al., 2018; Chen et al., 2019) pointed out these problems and studied the automated metrics in evaluating QA systems. Inspired by these works, we focus on studying and developing the evaluation metrics for genQA datasets that have more abstract and diverse answers. We analyze the problem of using existing n-gram similarity metrics across multiple genQA datasets and propose alternative metrics for genQA.

7 Conclusion

In this work, we study and improve the metrics for evaluating the correctness of answers in genQA, where the task is to generate an abstractive, free-form answer given a question and a context. We collected large-scale human judgements on two genQA datasets to compare the correlation with existing metrics. We show that existing metrics have a lower correlation with human judgement in the two datasets. We observe that previous n-gram-based similarity metrics cannot consider the importance weight of the words in the sentence. Based on this observation, we propose a new metric that can assign a weight to a word depending on the importance of evaluating the correctness. To compute this weight, we train KPQA that can predict the importance of words in a generated answer conditioned on the question. By adopting this weights for maximum similarity matching in several existing metrics, we propose KPQA-metric. Our new metric has a dramatically higher correlation with human judgements than existing metrics.

References

Huda Alamri, Chiiori Hori, Tim K Marks, Dhruv Batra, and Devi Parikh. 2018. Audio visual scene-aware dialog (avsd) track for natural language generation in dstc7. In DSTC7 at AAAI2019 Workshop, volume 2.

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 summarization, pages 65–72.

Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for generative multi-hop question answering tasks. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 4220–4230.

Anthony Chen, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Evaluating question answering evaluation. In Proceedings of the 2nd Workshop on Machine Reading for Question Answering, pages 119–124.

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.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In Advances in Neural Information Processing Systems, pages 13042–13054.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaqiaqiao She, et al. 2017. Dureader: a chinese machine reading comprehension dataset from real-world applications. arXiv preprint arXiv:1711.05073.
Drew A Hudson and Christopher D Manning. 2019. 
Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6700–6709.

Hyun Joon Jung and Matthew Lease. 2011. Improving consensus accuracy via z-score and weighted voting. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*.

Tomáš Kočínsky, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The narrativetqqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Klaus Krippendorff. 2011. *Computing krippendorff’s alpha-reliability*.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551.

J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, pages 159–174.

Hung Le, Doyen Sahoo, Nancy Chen, and Steven Hoi. 2019. Multimodal transformer networks for end-to-end video-grounded dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5612–5623.

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.

Chia-Wei Liu, Ryan Lowe, Iulian Vlad Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132.

Edward Loper and Steven Bird. 2002. NLTK: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, pages 63–70.

Preksha Nema and Mitesh M Khapra. 2018. Towards a better metric for evaluating question generation systems. *arXiv preprint arXiv:1808.10192*.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. *Choice*, 2640:660.

Kyosuke Nishida, Itsumi Saito, Kosuke Nishida, Kazutoshi Shinoda, Atsushi Otsuka, Hisako Asano, and Junji Tomita. 2019. Multi-style generative reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2273–2284.

Simon Ostermann, Michael Roth, Ashutosh Modi, Stefan Thater, and Manfred Pinkal. 2018. Semeval-2018 task 11: Machine comprehension using commonsense knowledge. In *Proceedings of the 12th International Workshop on semantic evaluation*, pages 747–757.

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 of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Andrew Shin, Yoshitaka Ushiku, and Tatsuya Harada. 2016. The color of the cat is gray: 1 million full-sentences visual question answering (fsvqa). *arXiv preprint arXiv:1609.06657*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Wikipedia. 2020. Likert scale — Wikipedia, the free encyclopedia. [http://en.wikipedia.org/w/index.php?title=Likert%20scale&oldid=951371208]. [Online; accessed 30-April-2020].

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R’emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface’s transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

An Yang, Kai Liu, Jing Liu, Yajuan Lyu, and Sujian Li. 2018. Adaptations of rouge and bleu to better evaluate machine reading comprehension task. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 98–104.

Jun Yin, Xin Jiang, Zhengdong Lu, Lifeng Shang, Hang Li, and Xiaoming Li. 2015. Neural generative question answering. *arXiv preprint arXiv:1512.01337*.

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

MS-MARCO MS-MARCO (Nguyen et al., 2016) is a large-scale machine reading comprehension dataset that provides ten candidate passages for each question. The model should consider the relevance of the passages for the given question and answer the question. One of the main features of this dataset is that it contains free-form answers that are abstractive. MS-MARCO provides two tasks, Natural Language Generation (NLG) task and QA task. For the NLG task, the model should generate an abstractive summary of the passages for given questions, which is a well-formed answer rather than an answer span in the passage. Although the QA task also provides some abstractive answers, most of the answers are short and do not contain the context or rationale of the question. Hence, we use the NLG subset of MS-MARCO dataset as a genQA dataset to study the metrics for genQA.

Audio Visual Scene-aware Dialog (AVSD) To study more general metrics for genQA, we also use a multimodal genQA dataset for our work. Audio Visual Scene-aware Dialog (AVSD) (Alamri et al., 2019) is a multimodal dialogue dataset composed of QA pair about Charades videos. Although the name of the dataset contains dialog, all of the dialog pairs are composed of questions answering about a video. The task of this dataset is to generate an answer for a question about a given video, audio, and the history of previous turns in the dialog. In other words, this task is to generate a free-form answer for a given multimodal context, which can be considered as a kind of genQA.

B Models

To investigate the algorithms for automatic metrics, we gather pairs of a sentence, \{answer candidate, true-answer\}. Note that each sentence is in natural language form. Collecting high-quality answer candidates for a given context and question is an essential step; thus, we choose models for each dataset from the latest research in the literature. We describe the models to generate the answer for two datasets we use in our work. For each dataset, we train the model that shows the highest performance in each dataset.

UniLM Since the publicly available code for state-of-the-art is not available for the NLG setting in MS-MARCO, we train the model with UniLM (Dong et al., 2019), which is a state-of-the-arts seq2seq model based on pre-trained representations from BERT (Devlin et al., 2019). UniLM, which stands for unified language model pretraining, is a pre-trained transformer network that can be easily fine-tuned for NLU and NLG. UNiLM achieves higher performance for various NLG tasks, such as abstractive summarization and question generation. We fine-tune UniLM for genQA similar to the way fine-tuning UniLM to NLG, where source sequences are each question and paragraphs, the target sequence is an answer. We add [SEP] tokens between the question and each paragraph. Then, we fine-tune UniLM for three epochs with this setting.

MTN For AVSD, we train the multimodal transformer network and is a transformer encoder-decoder framework (MTN) (Le et al., 2019), which is a state-of-the-art model for this task. MTN employs multimodal attention blocks to fuse multiple modalities such as text, video, and audio. We use the publicly available code to train the model with the trainset of this dataset. After training, we generate the answers for the testset released in the DSTC7 workshop (Alamri et al., 2018).

C Stopwords Removal

 Some of the words such as articles are commonly useless for measuring the correctness of the generated answer and even harm the evaluation by the unnecessary increase in the score. For the right example in Figure 1, the exact match in be and a results in a higher ROUGE or BLEU, although the answer is incorrect. Hence, we try to filter them when evaluating the correctness. We construct a stopword list based on the stopwords list for English in the NLTK library (Loper and Bird, 2002). We exclude tokens such as “no” and “not” in the original list since those words might be important words in genQA. By using our stopwords list, we filter stopwords in the generated sentence by setting their weights to zero when we calculate the metric. Then, we can give more focus to the remain words that have a higher possibility to be content words.