KPQA: A Metric for Generative Question Answering Using Keyphrase Weights

Hwanhee Lee[1], **Seunghyun Yoon**[1], **Franck Dernoncourt**[2],
Doo Soon Kim[2], **Trung Bui**[2], **Joongbo Shin**[1], **Kyomin Jung**[1],
[1] Dept. of Electrical and Computer Engineering, Seoul National University, Seoul, Korea
[2] Adobe Research, San Jose, CA, USA
{wanted1007, mysmilesh, jbshin, kjung}@snu.ac.kr
{franck.dernoncourt, dkim, bui}@adobe.com

Abstract

In the automatic evaluation of generative question answering (GenQA) systems, it is difficult to assess the correctness of generated answers due to the free-form of the answer. Moreover, there is a lack of benchmark datasets to evaluate the suitability of existing metrics in terms of correctness. To study a better metric for GenQA, we first create high-quality human judgments of correctness on two standard GenQA datasets. Using our human-evaluation datasets, we show that widely used n-gram similarity metrics do not correlate with human judgments. To alleviate this problem, we propose a new metric for evaluating the correctness of GenQA. Specifically, our new metric assigns different weights to each token via keyphrase prediction, thereby judging whether a generated answer sentence captures the key meaning of the reference answer. Our proposed metric shows a significantly higher correlation with human judgments than existing metrics in various datasets.

1 Introduction

Question answering (QA) has received consistent attention from the natural language processing community. Recently, research on QA systems has reached the stage of generating free-form answers, called GenQA, beyond extracting the answer to a given question from the context (Yin et al. 2016; Song, Wang, and Hamza 2017; Bauer, Wang, and Bansal 2018; Nishida et al. 2019; Bi et al. 2019, 2020). However, as a bottleneck in developing GenQA models, there are no proper automatic metrics to evaluate generated answers (Chen et al. 2019).

In evaluating a GenQA model, it is essential to consider whether a 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), that measure the word overlaps between the generated response and the reference answer; however, these metrics are insufficient to evaluate a GenQA system (Yang et al. 2018; Chen et al. 2019).

For instance, in the example in Figure 1 from the Microsoft MAchine Reading COmprehension (MS-MARCO) (Nguyen et al. 2016) dataset, the generated answer receives a high score on BLEU-1 (0.778) and ROUGE-L (0.713) due to the many overlaps of words with those in the reference. However, humans assign a low score of 0.063 on the scale from 0 to 1 due to the mismatch of critical information. In this example,

Context : ... , this process, called hypothesis testing, consists of **four steps**. , ...

Question : **How many steps** are involved in a hypothesis test?
Reference Answer : **Four steps** are involved in a hypothesis test.
Generated Answer : There are **seven steps** involved in a hypothesis test .

Human Judgment : 0.063
BLEU-1 : 0.778
ROUGE-L : 0.713
BLEU-1-KPQA : 0.057
ROUGE-L-KPQA : 0.127

Figure 1: An example from MS-MARCO (Nguyen et al. 2016) dataset where widely used n-gram similarity metrics does not align with human judgments of correctness.

to answer the question correctly, we expect the generated answer to include *four* in it since the question is asking about the number of steps. As in this example, we find that existing metrics often fail to capture the correctness of the generated answer that considers the key information in the question. The reason is that widely used n-gram based metrics equally consider the importance of each word when they evaluate answers.

To overcome this shortcoming of the existing metrics, we develop a novel method, Keyphrase Predictor for Question Answering (KPQA). KPQA computes the importance weight of each word in both the generated answer and the reference answer by considering the question. By integrating the output from the KPQA, we propose a *KPQA*-metric. 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*, *reference answer*} using a pre-trained KPQA; (2) We then compute a weighted similarity score by integrating the importance weights into existing metrics. The overall flow of our KPQA-metric is depicted in Figure 2. Our approach can be easily integrated into most existing metrics, including n-gram similarity metrics and the recently proposed BERTScore (Zhang et al. 2020).

To test our proposed metric, we create a benchmark dataset to evaluate automatic evaluation metrics for assessing the correctness of the generated response in GenQA. We collect new and high-quality human evaluations for the correctness of generated answers obtained from the models on two standard GenQA datasets: MS-MARCO and Audio Visual Scene-aware Dialog (AVSD) (Alamri et al. 2019).

We evaluate our proposed method on four datasets: MS-MARCO, AVSD, NarrativeQA (Kočiskỳ et al. 2018) and SemEval (Ostermann et al. 2018) datasets from (Chen et al. 2019). Our proposed metric shows significantly higher correlations with human judgments than previous metrics. For example, one of our KPQA integrated metrics, BERTScore-KPQA obtains Pearson correlation coefficients of 0.698 and 0.711 for MS-MARCO and AVSD, respectively, compared to the Pearson correlation coefficients of 0.469 and 0.592 for BERTScore.

We will release the human-annotated benchmark dataset and pre-trained models to compute the *KPQA*-metric to the research community[1].

2 Preliminaries: Automated Text Evaluation Metrics

We briefly review the current automated text evaluation metrics that have been used to evaluate 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 varies from 1 to 4, and the scores for varying n are aggregated with a geometric mean. In this work, we use 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 longest common subsequence between a candidate and the reference. Unlike BLEU, ROUGE-L has the advantage of not requiring a 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.

CIDER (Vedantam, Lawrence Zitnick, and Parikh 2015) is a consensus-based evaluation metric that is designed for a high correlation with human judgment in the image captioning problem. The most notable difference among above n-gram based metrics is that CIDEr uses Term Frequency-Inverse Document Frequency (TF-IDF) weights for human-like evaluation.

BERTScore is a recently proposed text evaluation metric that use pre-trained representations from BERT (Devlin et al. 2019). BERTScore first computes the contextual embeddings for given references and candidates independently with BERT, and then computes pairwise cosine similarity scores. When computing similarity, BERTScore adopts Inverse Document Frequency (IDF) to apply importance weighting.

[1] https://github.com/hwanheelee1993/KPQA

Dataset	Answer Length (avg.)	# Samples
MS MARCO	16.6	183k
AVSD	9.4	118k
Narrative QA	4.7	47k
SemEval	2.5	14k

Table 1: Statistics of the generative question answering dataset.

3 Collecting Human Judgments

3.1 Generating Answers

GenQA Datasets: To evaluate GenQA metrics, it is necessary to measure the correlation between human judgments and automated text evaluation metrics for evaluating the model generated answers. Recently, Chen et al. (2019) released human judgments of correctness for two GenQA datasets, NarrativeQA (Kočiskỳ et al. 2018) and SemEval-2018 Task 11 (SemEval) (Ostermann et al. 2018). However, we find that the average lengths of the answer sentence are 4.7 and 2.5 for NarrativeQA and SemEval, 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. We argue that evaluating long and abstractive answers is more challenging and suitable for studying the metrics for general form of GenQA. To fill this gap, we collect the human judgments of correctness for model generated answers on two other GenQA datasets, MS-MARCO (Nguyen et al. 2016) and AVSD (Alamri et al. 2019), which have longer answers than NarrativeQA and SemEval as shown in Table 1. For the MS-MARCO dataset, we use the Natural Language Generation (NLG) subset, which has more abstractive and longer answers than the Q&A subset.

GenQA Models: For each of the two datasets, we first generate answers for questions on evaluation sets using two trained GenQA models: UniLM (Dong et al. 2019) and MH-PGM (Bauer, Wang, and Bansal 2018) for MS-MARCO, MTN (Le et al. 2019) and AMF (Alamri et al. 2018; Hori et al. 2017) for AVSD. Details on these QA models are in Appendix. After answer generation, we randomly select 500 samples from each dataset. During the sampling, we discard samples if one of two GenQA models exactly generates the ground-truth answer since human evaluation is useless. In the following section, we detail the procedure for collecting human judgments.

3.2 Collecting Human Judgments of Answer Correctness

Generated responses are evaluated by humans to annotate the correctness of the generated answers compared to ground-truth answers. We hire workers from the Amazon Mechanical

Dataset - Model	α	# Annotators (avg.)
MS MARCO - UniLM	0.866	7.45
MS MARCO - MHPGM	0.742	6.71
AVSD - MTN	0.784	7.22
AVSD - AMF	0.814	7.27

Table 2: Inter annotator agreement measured by Krippendorff's alpha(α) and the average of number of annotators for each dataset.

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. We ask the workers to annotate correctness using a 5-point Likert scale (Likert 1932), where 1 means completely wrong, and 5 means completely correct.

Filtering Noisy Workers: Some workers did not follow the instructions, producing poor-quality judgments. To solve this problem, we filter noisy ratings using the z-score, as in (Jung and Lease 2011). We first compute the z-score among the ten responses for each sample. Then, we consider the responses whose z-score is higher than 1 to be noise and remove up to five of them in the order of the z-score. As a result, all of the samples have at least five annotations after removing 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 Krippendorff's α is higher than 0.6 for both datasets and models after filtering, as shown in Table 2. These coefficient numbers indicate a "substantial" 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 GenQA, we first propose KPQA. By considering the question, the KPQA assigns different weights to each token in the answer sentence such that salient tokens receive a high value. We then integrate the KPQA into existing metrics to make them evaluate correctness as well.

4.1 KPQA

For GenQA, we observe that each word has different levels of importance when assessing a generated answer. As shown in Figure 1, there exist keywords or keyphrases that are considered significant when evaluating the correctness of the answer. Additionally, some words, such as function words are mostly irrelevant to the correctness of the answer. Inspired by this observation, we introduce KPQA, which can predict the importance of each word when evaluating GenQA systems.

As shown in Figure 3, KPQA is a BERT-based (Devlin et al. 2019) classifier that predicts 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) , which consist of $\{$*passage*, *question*, *answer-span*$\}$. These datasets are transformed into pairs of $\{$*answer-sentence*, *question*, *answer-span*$\}$, where the answer-sentence is extracted from the passage so that it contains answer-span in it. The $\{$*question*, [SEP], *answer-sentence*$\}$ is fed into the KPQA to classify the answer-span, which is a set of salient tokens, in the given answer-sentence considering the question.

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 a pre-trained KPQA to get the importance weights that are useful for evaluating the correctness of generated answers in GenQA. The overall flow of our KPQA-metric is described in Figure 2. We describe how we combine these weights with existing metrics to derive the *KPQA*-metric.

We first compute the importance weights for a given question Q = (q_1, ..., q_l), reference answer X = (x_1, ..., x_n) and generated answer $\hat{X}$ = ($\hat{x}_1$, ..., $\hat{x}_m$) using pre-trained KPQA. We provide each pair $\{$*question*, *generated answer*$\}$ and $\{$*question*, *reference answer*$\}$ to pre-trained KPQA and get the output of the softmax layer. We define these parts as KeyPhrase Weight (KPW) as shown in Figure 3. We note that KPW$^{(Q,\hat{X})}$ = (w_1, ..., w_m) is an importance weight of generated answer $\hat{X}$ for a given question Q. These weights reflect the importance of each token for evaluating the correctness. We then compute *KPQA*-metric by incorporating the KPW into several existing metrics modifying the precision and recall to compute the weighted similarity.

BLEU-1-KPQA: We derive BLEU-1-KPQA, which is an weighted precision of unigram ($P^{KPQA}_{Unigram}$) as follows:

$$P^{KPQA}_{Unigram} = \frac{\Sigma^m_{i=1}\Sigma^n_{j=1}\text{KPW}^{(Q,\hat{X})}_i \cdot I(i,j)}{\Sigma^m_{i=1}\text{KPW}^{(Q,\hat{X})}_i}, \quad (1)$$

where $I(i,j)$ is an indicator function assigned the value of 1 if token x_i is the same as $\hat{x}_j$ and 0 otherwise.

ROUGE-L-KPQA: We also derive ROUGE-L-KPQA, which is a modified version of ROUGE-L using KPW to compute weighted precision(P^{KPQA}_{LCS}), recall(R^{KPQA}_{LCS}) and F1($F1^{KPQA}_{LCS}$), as follows:

$$P^{KPQA}_{LCS} = \frac{LCS^{KPQA}(X,\hat{X})}{\Sigma^m_{i=1}\text{KPW}^{(Q,\hat{X})}_i}, \quad (2)$$

$$R^{KPQA}_{LCS} = \frac{LCS^{KPQA}(X,\hat{X})}{\Sigma^n_{i=1}\text{KPW}^{(Q,X)}_i}, \quad (3)$$

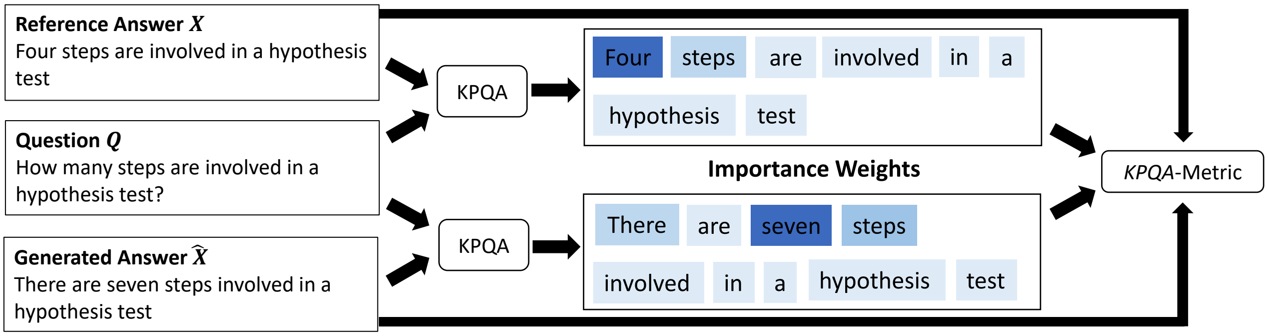

Figure 2: Overall flow of KPQA-metric. Importance weights are computed by pre-trained KPQA for each question-answer pair. And then these weights are integrated into existing metrics to compute weighted similarity.

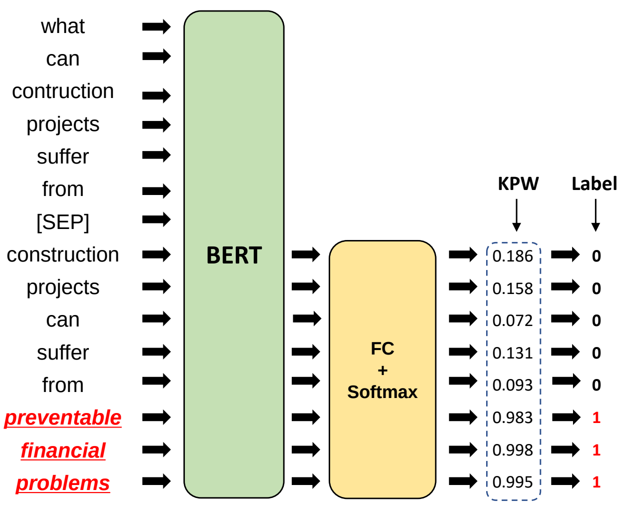

Figure 3: Overall architecture of KPQA and an output example of KPW from KPQA. The task for KPQA is to classify whether each word in the sentence is in the answer span for a given question. We train KPQA with the extractive QA dataset and use the output probability KPW when integrating it to KPQA-metric.

$$F_{LCS}^{KPQA} = \frac{(1+\beta^2) R_{LCS}^{KPQA} P_{LCS}^{KPQA}}{R_{LCS}^{KPQA} + \beta^2 P_{LCS}^{KPQA}}, \tag{4}$$

where LCS is the Longest Common Subsequence between a generated answer and a reference answer. The $LCS^{KPQA}(X, \hat{X})$ is defined as follows:

$$LCS^{KPQA}(X, \hat{X}) = \Sigma_{i=1}^{m} I_i \cdot \mathrm{KPW}_i^{(Q,\hat{X})}, \tag{5}$$

where I_i is an indicator function which is 1 if each word is in the LCS and 0 otherwise. β is defined in (Lin 2004).

BERTScore-KPQA Similar to ROUGE-L-KPQA, we compute BERTScore-KPQA using KPW. We first compute contextual embedding $\hat{\mathbf{x}}$ for generated answer $\hat{X}$ and $\mathbf{x}$ for reference X using the BERT model. Then, we compute weighted precision(P_{BERT}^{KPQA}), recall(R_{BERT}^{KPQA}) and F1($F1_{BERT}^{KPQA}$) with contextual embedding and KPW of each token as follows:

$$P_{BERT}^{KPQA} = \frac{\Sigma_{i=1}^{m} \mathrm{KPW}_i^{(Q,\hat{X})} \cdot \max_{x_j \in x} \mathbf{x_i}^T \hat{\mathbf{x}}_\mathbf{j}}{\Sigma_{i=1}^{m} \mathrm{KPW}_i^{(Q,\hat{X})}} \tag{6}$$

$$R_{BERT}^{KPQA} = \frac{\Sigma_{i=1}^{n} \mathrm{KPW}_i^{(Q,X)} \cdot \max_{\hat{x}_j \in \hat{x}} \mathbf{x_i}^T \hat{\mathbf{x}}_\mathbf{j}}{\Sigma_{i=1}^{n} \mathrm{KPW}_i^{(Q,X)}} \tag{7}$$

$$F1_{BERT}^{KPQA} = 2 \cdot \frac{P_{BERT}^{KPQA} \cdot R_{BERT}^{KPQA}}{P_{BERT}^{KPQA} + R_{BERT}^{KPQA}} \tag{8}$$

5 Experiments

5.1 Implementation Details

We choose two datasets SQuAD v1.1 (Rajpurkar et al. 2016), and MS-MARCO Q&A subset to train KPQA. We combine the training set of the two datasets and use a 9:1 split to construct the training and development set of KPQA.

For model parameters, we choose *bert-base-uncased* variants for the BERT model and use one fully-connected layer with softmax layer after it. We train 5 epochs and choose the model that shows the minimum evaluation loss. We provide more details in Appendix.

5.2 Results

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 judgments of correctness. We test using MS-MARCO, AVSD, from which we collected human judgments, and NarrativeQA and SemEval datasets from (Chen et al. 2019).

Performance Comparison: We present the correlation scores for the baseline metrics and KPQA-augmented ones for multiple datasets in Tables 3 and 4. The correlations between human judgment and most of the existing metrics such

Dataset	MS-MARCO				AVSD			
Model	UniLM		MHPGM		MTN		AMF	
Metric	r	ρ	r	ρ	r	ρ	r	ρ
BLEU-1	0.369	0.337	0.331	0.312	0.497	0.516	0.670	0.593
BLEU-4	0.173	0.224	0.227	0.26	0.441	0.492	0.600	0.570
ROUGE-L	0.317	0.289	0.305	0.307	0.510	0.528	0.656	0.581
METEOR	0.431	0.408	0.425	0.422	0.521	0.596	0.639	0.608
CIDEr	0.261	0.256	0.292	0.289	0.509	0.559	0.644	0.608
BERTScore	0.469	0.445	0.466	0.472	0.592	0.615	0.703	0.644
BLEU-1-KPQA	0.729	**0.676**	0.615	0.573	0.697	0.691	0.708	0.647
ROUGE-L-KPQA	**0.735**	0.674	**0.673**	0.631	0.674	0.677	0.708	**0.672**
BERTScore-KPQA	0.698	0.66	0.666	**0.662**	**0.711**	**0.699**	**0.718**	0.667

Table 3: Pearson Correlation(r) and Spearman's Correlation(ρ) between various automatic metrics and human judgments of correctness for MS-MARCO dataset and AVSD dataset. We generate the answers and collect human judgments for two models on each dataset. All of the results are statistically significant (p-value < 0.01).

Metric	NarrativeQA		SemEval	
	r	ρ	r	ρ
BLEU-1	0.634	0.643	0.359	0.452
BLEU-4	0.258	0.570	-0.035	0.439
ROUGE-L	0.707	0.708	0.566	0.580
METEOR	0.735	0.755	0.543	0.645
CIDEr	0.648	0.710	0.429	0.595
BERTScore	**0.785**	0.767	0.630	0.602
BLEU-1-KPQA	0.709	0.692	0.358	0.47
ROUGE-L-KPQA	0.767	0.746	0.729	0.675
BERTScore-KPQA	0.783	**0.769**	**0.748**	**0.677**

Table 4: Pearson Correlation(r), Spearman's Correlation(ρ) between various automatic metrics and human judgments of correctness for NarrativeQA dataset and SemEval 2018 Task 11 dataset. We use the dataset released from (Chen et al. 2019). All of the results are statistically significant (p-value < 0.01).

Dataset	MS-MARCO	
Model	UniLM	MHPGM
Metric	r	r
BLEU-1-KPQA	0.729	0.615
ROUGE-L-KPQA	**0.735**	**0.673**
BERTScore-KPQA	0.698	0.666
BLEU-1-KP	0.692	0.532
ROUGE-L-KP	0.710	0.627
BERTScore-KP	0.685	0.632
BLEU-1-KPQA$_{/SQuAD}$	0.645	0.532
ROUGE-L-KPQA$_{/SQuAD}$	0.642	0.572
BERTScore-KPQA$_{/SQuAD}$	0.633	0.595

Table 5: Ablation studies for our proposed metrics.

as BLEU or ROUGE-L are very low, and this shows that those widely used metrics are not adequate to GenQA. Moreover, the performance of existing metrics is especially low for the MS-MARCO dataset, which has longer and more abstractive answers than the other three datasets.

We observe a significantly higher correlation score for our proposed KPQA-metric compared to existing metrics. In the MS-MARCO dataset, our proposed metric significantly overwhelms existing metrics. For the NarrativeQA dataset, where existing metrics also have higher correlations, the gap in performance between KPQA-metric and existing metrics is low. We explain this is because the answers in NarrativeQA datasets are often a single word or short phrases that are already keyphrases.

Additionally, the dataset we collect has human judgments on a generated answer from two models for each dataset; thus we can observe how the performance of each metric depends on the type of GenQA model. The experimental results show that our proposed metric outperforms other metrics in both of the GenQA models for each dataset.

Comparison with IDF: The next best metric after our proposed metric is the original BERTScore, which uses contextual embeddings and adopts IDF based importance weighting. Since IDF is dependent on the word-frequency among the documents, it can assign a lower weight to some important words to evaluate correctness if they frequently occur in the corpus. On the other hand, our KPQA integrated metric assigns weights to words in the answer sentence using the context of the question. This approach provides dynamic weights for each word that leads to a better correlation with human evaluation as shown in Tables 3 and 4.

5.3 Ablation Study

Using the Question Context: Our KPQA uses the question as an additional context to predict the keyphrases in the sentence, as shown in Figure 3. To examine the power of utilizing the question information for the keyphrase predictor, we remove the question part from the dataset and train the keyphrase prediction model. With the newly trained

Context	... , it can take 5-20 hours of walking to lose 1 pound ... , ...
Question	How long do i need to walk in order to loose a pound ?
Reference	Walk for 5 to 20 hours to lose 1 pound .
IDF	Walk for 5 to 20 hours to lose 1 pound .
KPW	Walk for 5 to 20 hours to lose 1 pound .
	Human Judgment: 0.94, **BERTScore**: 0.72, **BERTScore-KPQA**: 0.93
UniLM	You need to walk for 5 to 20 hours in order to loose a pound .
IDF	You need to walk for 5 to 20 hours in order to loose a pound .
KPW	You need to walk for 5 to 20 hours in order to loose a pound .
	Human Judgment: 0.40, **BERTScore**: 0.48, **BERTScore-KPQA**: 0.51
MHPGM	You need to walk in order in order for 30 to 60 minutes per day
IDF	You need to walk in order in order for 30 to 60 minutes per day
KPW	You need to walk in order in order for 30 to 60 minutes per day

Table 6: An example of the scores given by humans, BERTScore and BERTScore-KPQA for the samples from MS-MARCO dataset. BERTScore uses IDF and BERTScore-KPQA uses KPW as importance weights to compute score. Heat map shows IDF and KPW, which are normalized between 0 and 1.

model, we compute the importance weights for words in the target sentence and apply them to BLEU-1, ROUGE-L, and BERTScore. We call this metric as "-KP" and report the results in Table 5. We observe that "-KPQA" metric is better than "-KP" metric for all of the three variants. These results show that training keyphrase predictor to find the short answer candidate in the sentence is effective for capturing the key information in the generated answer, but it is more effective when the question information is integrated.

Domain Difference: Our KPQA metric is a model-based metric; thus our proposed method may suffer from a domain shift problem. Although our metric is evaluated on out-of-domain datasets, we further examine the effect of the domain difference by changing the trainset of KPQA. Since we train KPQA with the combination of SQuAD and MS-MARCO Q&A dataset, the original KPQA works as in-domain for MS-MARCO. To measure the negative domain effect, we train KPQA with only SQuAD dataset and measure the performance of BERTScore-KPQA on MS-MARCO dataset. We annotate it "-$\text{KPQA}_{/\text{SQuAD}}$" and report the results in Table 5. This drop shows the effect of the negative domain shift for our KPQA-metric. However, "-$\text{KPQA}_{/\text{SQuAD}}$" is still much higher than all of the previous metrics.

5.4 Analysis

Correlation Among Question Type: Since MS-MARCO dataset provides the question type information (*PERSON*, *NUMERIC*, *DESCRIPTION*, *LOCATION*, *ENTITY*) for each {*question*, *answer*} pair, we evaluate the various metrics by the question type. We split the dataset into these five question types and measure the performance of various metrics with Pearson correlation coefficients for the generated answer from UniLM. As shown in Figure 4, our KPQA-metric

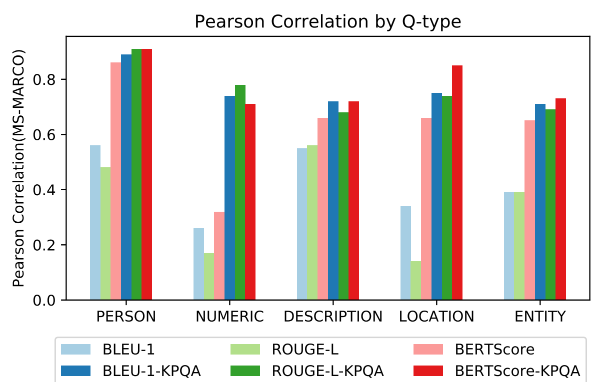

Figure 4: Pearson correlation coefficient among question types on MS-MARCO dataset.

variants outperform their original version in all of the question types. The performance gap is especially higher for the *NUMERIC* question type, whose answer sentence often has shorter keyphrase such as a number. For *ENTITY* and PERSON question types, the gap between KPQA-integrated metric and original metric is lower for BERTScore. We speculate that this is because the original BERTScore uses IDF based importance weighting unlike other metrics.

Rank-Pair: One practical usage of the text evaluation metric is ranking outputs of multiple models. Using the collected human judgments of correctness for the same 500 {*question*, *reference answer*} pairs for two models on MS-MARCO and AVSD, we can compare the output of each models through the human-annotated score. As shown in Table 6, humans judge

Metrics	MS-MARCO	AVSD
BLEU-1	63.44	72.02
ROUGE-L	61.29	70.98
BERTScore	67.74	78.24
BLEU-1-KPQA	74.19	**81.35**
ROUGE-KPQA	**76.34**	77.20
BERTScore-KPQA	**76.34**	**81.35**

Table 7: The percentage of matches at which human judgment and various metrics on ranking two models' output.

that the generated answer from UniLM is better than that from MHPGM (0.94 to 0.40). Similarly, both BERTScore and BERTScore-KPQA rank the output of UniLM's output as superior. To see the alignment of ranking ability among the various metrics with that of human judges, we conduct a "win-lose match" experiment, counting the number of times that a metric ranks the output of two models as the same as human judges. To prepare test samples, we chose only those whose gap between human judgment scores on the two models is greater than 2. Finally, we obtain 93 and 193 samples for MS-MARCO and AVSD, respectively. Considering that the range of scores is 1-5, this approach ensures that each output of the models has a clear quality difference. Table 7 shows the percentage of rank-pair matches for each metric with human judgments of correctness on two datasets. Our KPQA-metric shows more matches than previous metrics in all of the datasets; thus, it is more useful for comparing the generated answers from different models.

Multiple Sentence Answers: In GenQA, the answers can be multiple sentences like (Fan et al. 2019; Kwiatkowski et al. 2019). To verify our KPQA-metric on multiple sentence answers, we collect additional 100 human judgments for the generated answer using UniLM for the samples whose answers are multiple sentences in the MS-MARCO dataset. In these samples, BERTScore-KPQA obtains Pearson Correlation coefficient of 0.774 with human judgments while the original BERTScore obtains Pearson correlation coefficients of 0.712. This result shows that our metric can also be applied to multiple sentences. We present detailed results, including other metrics in the Appendix.

Error Analysis: We pick 100 error cases from MS-MARCO dataset in the order of a large difference in ranks among 500 samples between human judgments and BERTScore-KPQA. The importance weights have no ground-truth data; thus we manually visualize the weights as shown in Table 6 and analyze the error cases.

From the analysis, we observe some obvious reasons for the different judgments between humans and BERTScore-KPQA. We first classify error cases by the question types and observe that 51 cases belong to *NUMERIC*, and 31 cases belong to *DESCRIPTION*. We further analyze the *NUMERIC* question type and find that many parts of the errors are due to higher weights on units such as "million" or "years." There exist a total of ten error cases for this type, and we believe that there is room for improvement with regard to these errors through post-processing. In the case of the *DESCRIPTION* question type, 17 out of 31 cases are due to inappropriate importance weights. We speculate this result is because the keyphrases for the answers to questions belonging to the *DESCRIPTION* type are sometimes vague; thus, the entire answer needs to be considered when it is evaluated.

6 Related Work

One important next step for current QA systems is to generate answers in natural language for a given question and context. Following this interest, several generative (abstractive) QA datasets (Nguyen et al. 2016; He et al. 2017; Kočiský et al. 2018; Fan et al. 2019), where the answer is not necessarily in the passage, have recently been released. Since the task is to generate natural language for the given question, the QA system is often trained with seq2seq (Sutskever, Vinyals, and Le 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 were originally developed to evaluate machine translation and previous works (Liu et al. 2016; Nema and Khapra 2018; Kryscinski et al. 2019) have shown that these metrics have poor correlations with human judgments in other language generation tasks such as dialogue systems. As with other text generation systems, for GenQA, it is difficult to assess the performance through n-gram metrics. Especially, n-gram similarity metrics can give a high score to a generated answer that is incorrect but shares many unnecessary words with the reference answer. Previous works (Marton and Radul 2006; Yang et al. 2018; Chen et al. 2019) have pointed out the difficulty of similar problems and studied automated metrics for evaluating QA systems. Inspired by these works, we focus on studying and developing evaluation metrics for GenQA datasets that have more abstractive 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 paper, we create high-quality human judgments on two general forms of GenQA datasets, MS-MARCO and AVSD, and show that previous evaluation metrics are poorly correlated with human judgments in terms of the correctness of an answer. We propose KPQA-metric, which uses the pre-trained model that can predict the importance weights of words in answers to a given question to be integrated with existing metrics. Our approach has a dramatically higher correlation with human judgments than existing metrics, showing that our model-based importance weighting is critical to measure the correctness of a generated answer in GenQA.

References

Alamri, H.; Cartillier, V.; Das, A.; Wang, J.; Cherian, A.; Essa, I.; Batra, D.; Marks, T. K.; Hori, C.; Anderson, P.; et al. 2019. Audio visual scene-aware dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7558–7567.

Alamri, H.; Hori, C.; Marks, T. K.; Batra, D.; and Parikh, D. 2018. Audio visual scene-aware dialog (avsd) track for natural language generation in dstc7. In *DSTC7 at AAAI2019 Workshop*, volume 2.

Banerjee, S.; and Lavie, A. 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*, 65–72.

Bauer, L.; Wang, Y.; and Bansal, M. 2018. Commonsense for Generative Multi-Hop Question Answering Tasks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 4220–4230.

Bi, B.; Wu, C.; Yan, M.; Wang, W.; Xia, J.; and Li, C. 2019. Incorporating External Knowledge into Machine Reading for Generative Question Answering. 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)*, 2521–2530.

Bi, B.; Wu, C.; Yan, M.; Wang, W.; Xia, J.; and Li, C. 2020. Generating Well-Formed Answers by Machine Reading with Stochastic Selector Networks. In *AAAI*, 7424–7431.

Chen, A.; Stanovsky, G.; Singh, S.; and Gardner, M. 2019. Evaluating Question Answering Evaluation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, 119–124.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 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)*, 4171–4186.

Dong, L.; Yang, N.; Wang, W.; Wei, F.; Liu, X.; Wang, Y.; Gao, J.; Zhou, M.; and Hon, H.-W. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, 13042–13054.

Fan, A.; Jernite, Y.; Perez, E.; Grangier, D.; Weston, J.; and Auli, M. 2019. ELI5: Long Form Question Answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 3558–3567.

Gers, F. A.; Schmidhuber, J.; and Cummins, F. 2000. Learning to Forget: Continual Prediction with LSTM. *Neural Computation* 12(10): 2451–2471.

He, W.; Liu, K.; Liu, J.; Lyu, Y.; Zhao, S.; Xiao, X.; Liu, Y.; Wang, Y.; Wu, H.; She, Q.; et al. 2017. Dureader: a chinese machine reading comprehension dataset from real-world applications. *arXiv preprint arXiv:1711.05073* .

Hori, C.; Hori, T.; Lee, T.-Y.; Zhang, Z.; Harsham, B.; Hershey, J. R.; Marks, T. K.; and Sumi, K. 2017. Attention-based multimodal fusion for video description. In *Proceedings of the IEEE international conference on computer vision*, 4193–4202.

Jung, H. J.; and Lease, M. 2011. Improving consensus accuracy via z-score and weighted voting. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*.

Kočiský, T.; Schwarz, J.; Blunsom, P.; Dyer, C.; Hermann, K. M.; Melis, G.; and Grefenstette, E. 2018. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics* 6: 317–328.

Krippendorff, K. 2011. Computing Krippendorff's alpha-reliability .

Kryscinski, W.; Keskar, N. S.; McCann, B.; Xiong, C.; and Socher, R. 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)*, 540–551.

Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; et al. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics* 7: 453–466.

Landis, J. R.; and Koch, G. G. 1977. The measurement of observer agreement for categorical data. *biometrics* 159–174.

Le, H.; Sahoo, D.; Chen, N.; and Hoi, S. 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*, 5612–5623. URL https://www.aclweb.org/anthology/P19-1564.

Likert, R. 1932. A technique for the measurement of attitudes. *Archives of psychology* .

Lin, C.-Y. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, 74–81. Barcelona, Spain: Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W04-1013.

Liu, C.-W.; Lowe, R.; Serban, I. V.; Noseworthy, M.; Charlin, L.; and Pineau, J. 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*, 2122–2132.

Marton, G.; and Radul, A. 2006. Nuggeteer: Automatic Nugget-Based Evaluation using Descriptions and Judgements. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, 375.

Nema, P.; and Khapra, M. M. 2018. Towards a Better Metric for Evaluating Question Generation Systems. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* 3950–3959.

Nguyen, T.; Rosenberg, M.; Song, X.; Gao, J.; Tiwary, S.; Majumder, R.; and Deng, L. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *choice* 2640: 660.

Nishida, K.; Saito, I.; Nishida, K.; Shinoda, K.; Otsuka, A.; Asano, H.; and Tomita, J. 2019. Multi-style Generative Reading Comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2273–2284.

Ostermann, S.; Roth, M.; Modi, A.; Thater, S.; and Pinkal, M. 2018. Semeval-2018 task 11: Machine comprehension using commonsense knowledge. In *Proceedings of the 12th International Workshop on semantic evaluation*, 747–757.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 311–318. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics. doi:10.3115/1073083.1073135. URL https://www.aclweb.org/anthology/P02-1040.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2383–2392.

Song, L.; Wang, Z.; and Hamza, W. 2017. A unified query-based generative model for question generation and question answering. *arXiv preprint arXiv:1709.01058* .

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112.

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

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; and Brew, J. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv* abs/1910.03771.

Yang, A.; Liu, K.; Liu, J.; Lyu, Y.; and Li, S. 2018. Adaptations of ROUGE and BLEU to Better Evaluate Machine Reading Comprehension Task. In *Proceedings of the Workshop on Machine Reading for Question Answering*, 98–104.

Yin, J.; Jiang, X.; Lu, Z.; Shang, L.; Li, H.; and Li, X. 2016. Neural generative question answering. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, 2972–2978.

Zhang, T.; Kishore, V.; Wu, F.; Weinberger, K. Q.; and Artzi, Y. 2020. BERTScore: Evaluating Text Generation with BERT. In *International Conference on Learning Representations*. URL https://openreview.net/forum?id=SkeHuCVFDr.

Dataset	MS-MARCO	
Metric	r	ρ
BLEU-1	0.363	0.364
ROUGE-L	0.584	0.607
BERTScore	0.712	0.728
BLEU-1-KPQA	0.529	0.540
ROUGE-L-KPQA	0.642	0.648
BERTScore-KPQA	**0.774**	**0.786**

Table A1: Correlation coefficients between various automatic metrics and human judgments of correctness for multiple sentence inputs.

Question : How to cook sausage peppers onions ?
Reference Answer : To cook sausage peppers onions first place the sausage in a large skillet over medium heat, and brown on all sides after that remove from skillet, and slice meelt butter in the skillet, stir in the yellow onion, red onion, and garlic, and cook 2 to 3 minutes and then mix in red bell pepper and green bell pepper season with basil, and oregano in last stir in white wine.
Generated Answer : To cook sausage peppers onions , preheat the oven to 350 degrees fahrenheit . place the onions in the oven and cook for 20 minutes

Figure A1: An example from MS-MARCO (Nguyen et al. 2016) dataset where the answers are composed of multiple sentences.

A Additional Results

A.1 Results on Multiple Sentence Inputs

The answers for GenQA can be multiple sentences like (Fan et al. 2019). Hence, we further collect additional 100 samples whose answers are only composed of multiple sentences in MS-MARCO dataset like Figure A1 to check whether our KPQA-metric is robust on multiple sentence inputs. We first generate the answer for each sample through UniLM and collect the results of evaluating it to 10 people just like the other samples in the main result. Then, we compute correlation coefficient between human judgments and various metrics. As shown in Table A1, our KPQA integrated metric has higher correlation than other metrics for the sample that have multiple sentence inputs, similar to the the main result.

B Data Collection

B.1 Datasets

We collect human judgments of correctness for two GenQA datasets, MS-MARCO (Nguyen et al. 2016) and AVSD (Alamri et al. 2019). We describe the properties of each dataset in this section.

MS-MARCO MS-MARCO 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 Q&A 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 Q&A 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. Also, we use the training set of Q&A subset to train and evaluate KPQA, since most of the samples in this subset has exact answer spans in the passage like SQuAD.

Dataset	Model	BLEU-1	ROUGE-L
MS-MARCO	UniLM	60.2	63.1
	MHPGM	43.7	53.9
AVSD	MTN	67.3	52.6
	AMF	62.6	48.7

Table A2: Performance of the model we trained to generate answers

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) 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.2 Instructions to Annotators

The instructions to annotators in MTurk are shown in Figure A2.

B.3 Models

To investigate the performance of automatic metrics, we gather pairs of a sentence, {generated answer, *reference answer*}. Collecting high-quality answer candidates for a given context and question is an essential step; thus, we choose two models for each dataset from the latest research in the literature. We train two models UniLM (Dong et al. 2019) and MHPGM (Bauer, Wang, and Bansal 2018) for MS-MARCO dataset. For AVSD dataset, we train two models MTN (Le et al. 2019) and AMF (Alamri et al. 2018). We present the performance of each model we trained in Table A2. We briefly describe the models and the training details to generate the answer for two datasets.

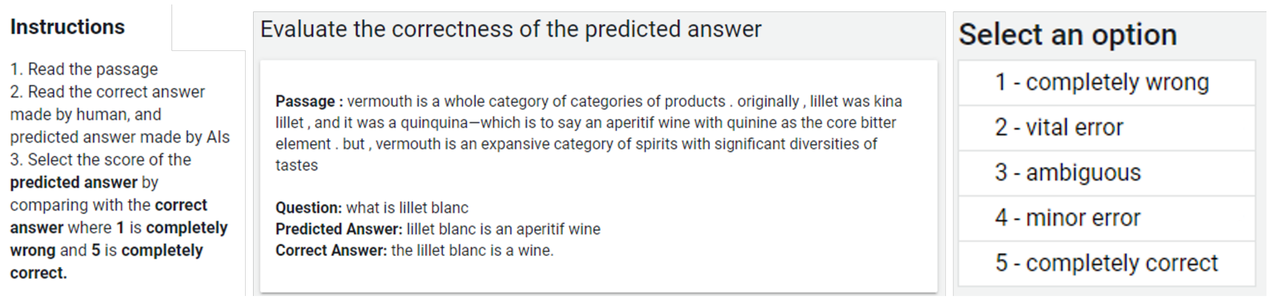

Figure A2: Instruction for MTurk workers

UniLM UniLM[2], which stands for unified language model pre-training, is a powerful seq2seq model based on pre-trained representations from BERT (Devlin et al. 2019). UniLM 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.

MHPGM MHPGM, which stands for multi-hop pointer generator network. MHPGM uses multi-hop reasoning QA model that can integrate commonsense information. This model uses pointer-generator decoder to generate the answer. We train the model for three epochs with batch size 24 using the public code[3].

MTN MTN (Le et al. 2019)[4], which is a multimodal transformer encoder-decoder framework, is a state-of-the-art model for AVSD. MTN employs multimodal attention blocks to fuse multiple modalities such as text, video, and audio. We train 10 epochs with batch size 256 and generate the answers for the testset released in the DSTC7 workshop (Alamri et al. 2018).

AMF AMF is an Attentional Multimodal Fusion based model (Hori et al. 2017) introduced as a baseline system for DSTC7 AVSD workshop (Alamri et al. 2018)[5], It is composed of RNN and multimodal attention architecture. This model encode the multimodal inputs with LSTM (Gers, Schmidhuber, and Cummins 2000) and fuse the information with modality-dependent attention mechanism. We train this model with 15 epochs with batch size 64.

[2]https://github.com/microsoft/unilm
[3]https://github.com/yicheng-w/CommonSenseMultiHopQA
[4]https://github.com/henryhungle/MTN
[5]https://github.com/dialogtekgeek/AudioVisualSceneAwareDialog

Dataset	F1
SQuAD	55.81
MS-MARCO Q&A	59.26

Table A3: Performance of our keyphrase predictor in development set of each dataset.

C Experimental Details

In this section, we describe some experimental details of our KPQA-metric.

C.1 Significant Test

For all of the correlation coefficients we computed in the paper, we use a t-test using a null hypothesis that is an absence of association to report p-value, which is the standard way to test the correlation coefficient.

C.2 Environment

For the experiments, we use Intel(R) Core(TM) i7-6850K CPU (3.60 GHz) with GeForce GTX 1080 Ti. The software environments are Python 3.6 and PyTorch 1.3.1.

C.3 KPQA Performance

We present the performance of KPQA on keyphrase prediction for evaluation data in Table A3.

C.4 BERTScore

For computing BERTScore we use *bert-large-uncased-whole-word-masking-finetuned-squad* variant from (Wolf et al. 2019)[6] 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 judgments 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. We compute all of the BERTScore including original BERTScore and BERTScore variants using this BERT model.

[6]https://github.com/huggingface/transformers