RT-VQ^{2}A^2: Real Time Vector Quantized Question Answering with ASR

Kyungho Kim, Seongmin Park, Jihwa Lee
ActionPower
{kyungho.kim, seongmin.park, jihwa.lee}@actionpower.kr

Abstract

In Spoken Question Answering (SQA), automatic speech recognition (ASR) outputs are often relayed to language models for QA. However, constructing such a cascaded framework with large language models (LLMs) in a real-time SQA setting involves realistic challenges, such as noise in the ASR output, the limited context length of LLMs, and latency in processing large models. This paper proposes a novel model-agnostic framework, ***RT-VQ^{2}A^2***, to address these challenges. RT-VQ^{2}A^2 consists of three steps: *codebook preparation*, *quantized semantic vector extractor*, and *dual segment selector*. We construct a codebook from clustering, removing outliers on a text corpus derived from ASR to mitigate the influence of ASR error. Extracting quantized semantic vectors through a pre-built codebook shows significant speed and performance improvements in relevant context retrieval. Dual segment selector considers both semantic and lexical aspects to deal with ASR error. The efficacy of RT-VQ^{2}A^2 is validated on the widely used Spoken-SQuAD dataset.

Keywords: spoken question answering, vector quantization, real time

1. Introduction

Question Answering (QA) (Rajpurkar et al., 2016; Trischler et al., 2017; Joshi et al., 2017; Chen et al., 2017) in text modality is called Machine Reading Comprehension (MRC) (Si et al., 2021). It has been studied as a fundamental Natural Language Understanding (NLU) (Devlin et al., 2019) task to evaluate how well a machine understands a given context and query. Therefore, lots of research on QA have been conducted, and show better performance beyond human capabilities in several tasks and environments (Rajpurkar et al., 2016, 2018; Koivisto and Grassini, 2023), .

Spoken Question Answering (SQA) (Lee et al., 2018, 2019; Menevşe et al., 2019) is a more challenging task as it requires the combination of an audio signal and natural language comprehension to provide appropriate answers to given audio documents and queries. Due to the nature of the SQA task, many studies have proposed multi-modal models to improve performance (Lin et al., 2022b; You et al., 2020; Chenyu et al., 2021). However, these models require complex design and additional training to handle multi-modal input. Furthermore, these complicated models are challenging to use in real-time Automatic Speech Recognition (ASR) environments due to high computational demands (Kuang et al., 2022).

In addition, errors, which occur while converting audio signals to text, make the SQA task even more challenging when using language model for QA. To address this issue, QA model is trained on poorly refined and error-prone text inputs (Sidiropoulos et al., 2022; Ravichander et al., 2021a; Lin et al., 2022a). (Sidiropoulos et al., 2022) analyzes the impact of speech recognition errors on QA. (Ravichander et al., 2021a) adopts classifying error by type and eliminating errors through post-hoc processing. (Lin et al., 2022a) decides to use the audio signal directly without text result for SQA. These studies are mainly focused on minimizing ASR errors through model finetuning and post-processing. However, our focus is considering the efficiency where error processing and QA are performed simultaneously within a limited time for real-time scenarios.

Another approach to solving SQA problems is to utilize Large Language Models (LLMs), which have recently rapid growth of attention to their promising performance (Raffel et al., 2019; Xue et al., 2021a; Chung et al., 2022; Ouyang et al., 2022). LLMs have led to significant improvements in performance on various Natural Language Processing (NLP) tasks, including QA. As a result of significant advancements in LLM, applying LLM to the audio transcript has become an attractive option (Chuang et al., 2019; Martínek et al., 2022; Higuchi et al., 2022). To this end, a cascade system converts a given audio document into text modality through the ASR module, followed by a text QA model through the LLM (Su and Fung, 2020).

However, serious challenges exist in directly applying a text QA model to the real-time SQA scenario. First, due to the time complexity of the attention mechanism and the limitation of GPU memory, the length of text input processed by the LLM may be shorter than the length of the audio document that the user wants to process. Second, the long processing time of LLM makes it challenging to use in real-time SQA scenarios, where QA is performed on the audio document with ASR.

LREC-COLING 2024, pages 14204–14214

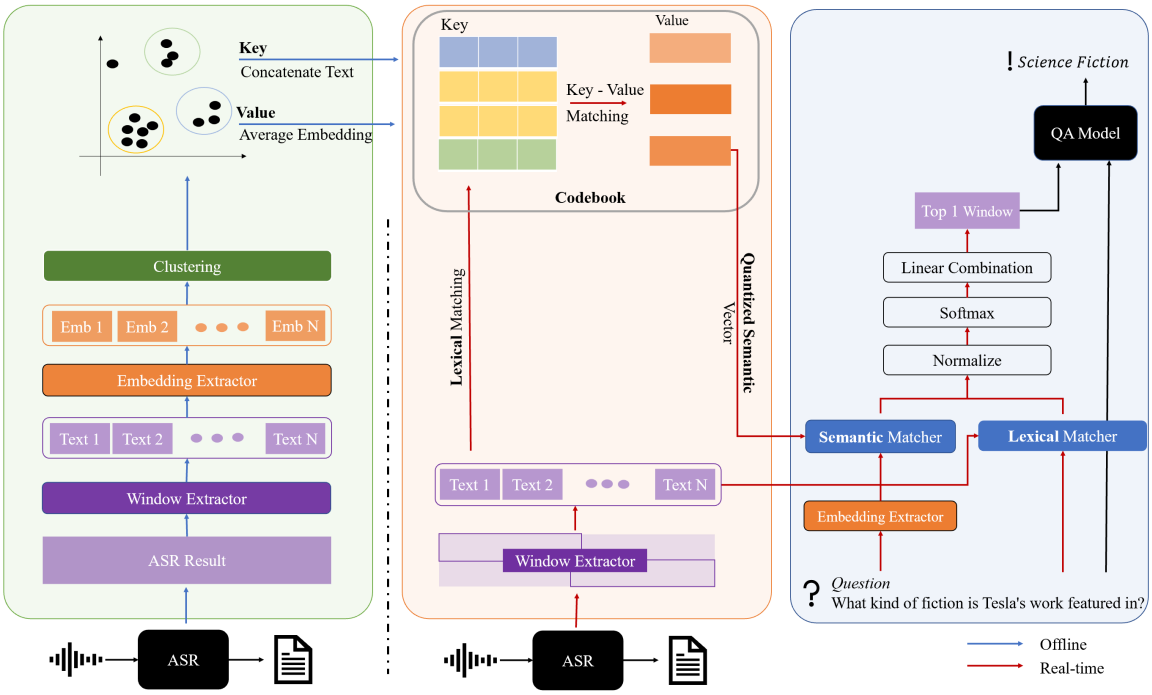

Figure 1: The overall framework of our proposed RT-VQ^{2}A^2.

To address these issues, it is necessary to extract relevant parts of the audio document to answer the question for a given query, similar to a search engine. Due to time constraints, searching is mainly done through lexical matching, which checks the degree of match between words or tokens. However, relying solely on lexical matching to find relevant parts is ineffective due to spelling or grammar errors in the ASR output. Therefore, conducting a semantics-reflecting search, called Dense Passage Retrieval (DPR) (Chen and Ren, 2020; Tang et al., 2022) based approach, is necessary. DPR utilizes the similarity of embeddings of documents. However, DPR assumes all documents are stored offline and calculates embeddings for all documents in advance. This makes DPR challenging to apply to real-time SQA scenarios where users request audio documents on the fly. (Kim et al., 2021) tries to reduce search time through query generation, but documents still need to be preprocessed in advance.

This paper proposes a novel framework called Real-Time Vector Quantized Question Answering (RT-VQ^{2}A^2) that enables fast semantic search through context quantization. Our research is model-agnostic and is orthogonal to developments in generative QA. Existing QA models can employ RT-VQ^{2}A^2 for fast and accurate context retrieval. In addition our proposed framework RT-VQ^{2}A^2 consists of three main steps. The first step is preparing a codebook in advance to enable real-time semantic vector extraction. In the second step, the codebook is utilized to obtain quantized semantic vectors for each segment, which is formed by splitting a document into multiple parts, in real-time. The final step takes both lexical and semantic aspects into account and extracts the audio segment most suitable for a given query.

As mentioned above, errors in ASR results and time-consuming semantic vector extraction are the biggest challenges in real-time SQA scenarios. To address this, we perform clustering on a text corpus consisting of ASR results from multiple audio documents when creating the codebook. By grouping similar topics, we expect that the influence of ASR error in individual documents would be mitigated in a larger text set which is composed of more data. Additionally, we experimentally discover that a clustering algorithm (Schubert and Gertz, 2018; Ankerst et al., 1999) that can remove outlier itself shows the best performance as evidence to support this approach. Furthermore, RT-VQ^{2}A^2 does not directly extract semantic vectors with an LM for all possible subsets of the given document. Instead, our approach returns quantized semantic vectors for each segment of the document through a pre-constructed codebook. The process of extracting quantized semantic vectors using both lexical matching and key-value matching of the codebook, as employed by RT-VQ^{2}A^2, shows a speed improvement of more than 10 times compared to directly extracting dense vectors using an LLM. The efficiency of our proposed method is demonstrated by the Spoken-SQuAD dataset (Lee et al., 2018), which is widely used in SQA. In particular, when the Word Error Rate (WER) is 22, our proposed quan-

tized semantic vector shows better performance and speed rather than using dense vectors.

In summary, our main contributions can be summarized as follows:

- We propose a novel framework, **RT-VQ^{2}A^2**, to address the challenges of applying LLM to SQA with ASR in real-time scenarios.
- We construct the **codebook** by clustering on a text corpus consisting of ASR results from multiple audio documents. It mitigates the influence of ASR error and experimentally demonstrates the importance of clustering to remove outliers.
- We design the process of extracting the **quantized semantic vector** using only the traditional lexical matching with the codebook to enable real-time semantic vector extraction for SQA task. It is faster and more efficient than directly extracting dense vectors by LLM.
- We introduce the **dual segment extractor** to get the small portion of an audio document, which is related to the given query, by considering both lexical and semantic aspects.

2. Preliminary

2.1. Spoken Question Answering

We represent the SQA dataset as a collection of N triplets $\mathbb{D} = \{S_i, Q_i, A_i\}_{i=1}^{N}$, where each triplet consists of spoken audio document $S_i \in \mathbb{S}$, question $Q_i \in \mathbb{Q}$, and answer $A_i \in \mathbb{A}$. The objective of the SQA task is to predict the gold answer distribution $\mathbb{A}$ with a given spoken audio document $\mathbb{S}$ and question $\mathbb{Q}$. A cascade system is built to achieve the goal by combining an ASR with a text-based QA model. A cascade system extracts the transcribed text results T_i for each audio document S_i by ASR and then inputs the text-question pair into the text QA model to obtain the final answer. This can be represented as follows:

$$\begin{aligned} T_i &= f_{asr}(S_i), \\ P(A_i|T_i, Q_i) &= f_{qa}(T_i, Q_i) \end{aligned} \tag{1}$$

where f_{asr} and f_{qa} refer to the ASR model and the text QA model, respectively.

2.2. Dense Passage Retrieval

Dense Pass Retrieval (DPR) is designed for the efficient retrieval of relevant information considering semantic information. It works by embeddings of text documents using neural network models, where similar vectors represent similar semantic meanings. This allows for more accurate and efficient retrieval of relevant passages or documents with a given query. It can be denoted as follows:

$$P = \underset{p_i}{\operatorname{argmax}}\ sim(E(q), E(p_i)) \tag{2}$$

where P denotes the passage recording the highest similarity among the given passages p_i for a given query q, and E denotes embedding from an LM.

However, DPR assumes that all documents are stored offline, and embeddings of documents are calculated in advance. This means that DPR may not be suitable for situations where new documents are constantly added, and high-speed calculation is required, such as real-time ASR.

2.3. Efficient QA

There exist studies aimed at improving the efficiency of Question Answering models, allowing for faster processing or operation on documents with longer contexts. DeQA (Cao et al., 2019) proposes a method for making QA models workable on mobile devices or in the cloud through neural encoding offloading and memory optimization. However, it focuses on small LMs, unlike the popular LLMs nowadays. The most similar aspects between our paper and DeQA are the use of a key-value database when extracting sentence semantic embeddings. In DeQA, static word embeddings are stored in a key-value database and the sentence embedding is obtained by averaging the embeddings of the words for memory optimization. However, in our real-time SQA scenario, directly extracting semantic vector by exact matching of lexical is impossible due to the possibility of words being misspelled by ASR errors. (Choi et al., 2017) aims to apply the QA model to long documents by selecting appropriate sentences using a reinforcement learning-based Bag-Of-Words(BOW) model. However, this model-based approach inevitably suffers from increased latency. Moreover, additional modules must be trained in comparison to conventional QA models. In contrast, our proposed method relies on the embeddings of LLMs, which are rapidly evolving and can be used off-the-shelf. Finally, we suggest an efficient SQA framework that considers ASR error, long context, and real-time constraints despite using LLMs.

3. Model

In this section, we introduce our architecture to utilize LLM with ASR model for real-time question answering. Figure 1 presents an overview of our approach, which introduces codebook preparation (Section 3.1), which is processed in advance, quantized semantic vector (Section 3.2) and dual segment extractor (Section 3.3) for real-time reaction.

3.1. Codebook Preparation

As shown in the green (left) block of Figure 1, we prepare a codebook using a collection of audio documents we possess in advance. Algorithm 1 presents the process of preparing the codebook as pseudo-code. First, ASR is performed for all audio documents to convert them into text. Then, we divide the resulting text into segments of appropriate size using the sliding window technique, which satisfies the input length limit of the model and enables fast processing by the language model. For each segment, a vector embedding reflecting semantic information is extracted by the encoder of LM. We conduct clustering on the embeddings to group segments with similar semantic meanings into one cluster. The average embedding of the segments in a cluster is used as the value of cluster in the codebook. The ASR results of all segments that make up a cluster are concatenated to form the key of the cluster in the codebook as text modality. The resulting codebook consists of key-value pairs, where each cluster corresponding to one key-value pair. During real-time question answering, the prepared codebook is used to obtain the semantic vector that matches the given segment quickly. The detail is described in the below Section 3.2.

Algorithm 1 Codebook Preparation

Input: audio set $\mathbb{A}$
Output: codebook C

▷ make codebook with pre-collected audio set
$text \leftarrow ASR(\mathbb{A})$
$segments \leftarrow sliding\ window(text)$
$embeddings \leftarrow encoder(segments)$
$clusters \leftarrow clustering(embeddings)$
$C \leftarrow \{\}$
for $cluster$ in $clusters$ **do**
    ▷ create key-value pair for each cluster
    $key \leftarrow []$
    $value \leftarrow []$
    for $segment$ in $cluster$ **do**
        $key \leftarrow key + text\ of\ segment$
        $value \leftarrow value + emb.\ of\ segment$
    end for
    $value \leftarrow average(value)$
    $C[key] \leftarrow value$
end for
return C ▷ return codebook

3.2. Quantized Semantic Vector Extractor

As shown in the orange (middle) block of Figure 1, this module is designed to enable real-time response by quickly obtaining the semantic vector of each segment that makes up the audio document when a user requests audio documents and question answering. To achieve this purpose, the offline-prepared codebook is used. As mentioned above, the codebook consists of key-value pairs, where the key is the text modality and the value is a high-dimensional vector embedding modality that can contain semantic information. In addition, text modality has the feature of performing retrieval at high speed through lexical matching. From this point of view, lexical matching is performed between the key of the codebook and the ASR result of each audio document segment to find the most similar codebook block for each segment. Then, through key-value matching of the codebook, the value of the codebook block is considered as the semantic vector of the corresponding segment. The kind of semantic vectors obtained for each segment is limited to the number of key-value pairs in the codebook. Therefore, we call this module the quantized semantic vector extractor. The formal representation of the above process is as follows:

$$\begin{aligned} K &= \underset{C_{key}}{\operatorname{argmax}}\ f_{lex}(f_{asr}(s), C_{key}), \\ V &= C[K] \end{aligned} \tag{3}$$

where f_{lex} refers the score of lexical matching that represents relevance of given text, s refers to segment of audio document, C refers to codebook, and V means the value of codebook block, which is treated as a quantized semantic vector of the corresponding segment.

3.3. Dual Segment Extractor

In the blue (right) block of Figure 1, we select one segment from multiple segments that make up an audio document, which has the best matching score with the query, considering both the lexical matching score and the semantic matching score. The semantic vector of the segment used in the semantic matching process is extracted from the quantized semantic vector extractor. Also, since the score ranges of lexical and semantic matching are different, each score is normalized and passed through a softmax layer. The final score is calculated by a linear combination of lexical matching score and semantic matching score. Finally, we select the top 1 segment, based on the final score along with the query, as input of the QA model to generate the final answer. This process can be expressed as follows:

$$\begin{aligned} Score_{lex} &= \text{Softmax}(\text{Norm}(f_{lex}(f_{asr}(s), q))), \\ Score_{sem} &= \text{Softmax}(\text{Norm}(f_{sem}(V, E(q))) \end{aligned} \tag{4}$$

where q refers the question, and f_{sem} is the score of semantic matching obtained from the similarity of semantic vector.

The final score can be denoted as follows:

$$Score_{fin} = \alpha * Score_{lex} + (1 - \alpha) * Score_{sem} \tag{5}$$

where $\alpha \in [0, 1]$ is a hyper-parameter that controls the weight between the lexical and semantic matching scores.

4. Experiment

Dataset	# Trainset	# Devset	# Cluster
WER 54	4069	1282	27
WER 44	3581	1770	51
WER 22	4067	1284	1151

Table 1: The statistic of each dataset on Spoken-SQuAD.

4.1. Dataset

To demonstrate the performance of our proposed framework, we conduct experiments using the Spoken-SQuAD dataset Lee et al. (2018). Spoken-SQuAD is a dataset created by converting the existing SQuAD QA dataset Rajpurkar et al. (2016) through text-to-speech and then transcribing it back into text using ASR. The test set of the Spoken-SQuAD dataset consists of three versions, each composed of a different Word Error Rate (WER). Spoken-SQuAD obtains two different WERs by adding two different levels of white noise add into the audio files of testing set. In order to maintain consistency about WER between the codebook preparation and inference step, we split the given test set into a train and dev set. The train set is used for codebook preparation, while the dev set is used for performance evaluation. In other words, our experiment is the same as using three ASR models with the same speech document but different performances.

The criteria for splitting the dataset is set based on the characteristics of the QA model and the ASR result. Since the generation-based LM heavily relies on the context of the input document In other words, the documents, which do not contain the exact answer words due to ASR error, do not adequately measure the performance. Therefore, we decide to use those documents, lacking correct answer words, for code preparation. However, in WER 22 version, all answer words existed accurately in the documents. In this case, to maintain consistency in our experiments, we adjust the number of train and dev sets used in WER 22 to a similar level as in WER 54 with specific random seed to reproducibility. Additionally, the number of clusters is automatically selected by the algorithm. Table 1 shows the statistics of the datasets used in our experiments. One thing to note is that the number of clusters varies greatly for each dataset. This is because we set the different minimum number of samples which is required to configure the cluster for each data set. This will be analyzed in Section 6.1 and Section 6.3.

4.2. Backbone Model

FLAN-T5-Large (Chung et al., 2022) is utilized as the LLM to extract semantic vectors and perform question answering. There are several reasons why we use FLAN-T5 [1] as our backbone instead of existing SQA models. First, the scenarios that existing SQA models consider are different from the scenario that our model aims to achieve. Existing SQA models, such as SpeechBERT (Chuang et al., 2019), use audio signals in conjunction with ASR results. However, the scope of this research is focused on QA soley in the text modality, and concerns scenarios where QA model is applied cascade manner from the ASR. Second, widely known SpeechBERT or distilation-based models (You et al., 2022) are not publicly available, which would make it difficult to reproduce our results. Third, although SpeechBERT and FLAN-T5 target different main scenario, FLAN-T5 shows similar results (or slightly better results) in all WERs in the cascade setting. For these reasons, we select FLAN-T5 as our backbone model.

4.3. Experiment Detail

The experiment is conducted using a single NVIDIA RTX A6000 GPU. For sliding window, both window size and stride are set to 256 to slice the document into multiple windows. To perform QA using FLAN-T5 without fine-tuning, the prompt is set to “answer question”, and during the process of extracting semantic vectors, only the text of the given segment is used as input without any prompt. In DPR, FLAN-T5 with same settings is utilized to extract dense semantic vector.

The lexical matcher is set to BM25 (Robertson and Zaragoza, 2009), which is widely used as a ranking function in search engines that calculates the relevance score between a query and a document based on term frequency, document length, and inverse document frequency. The semantic matching score is calculated by the cosine similarity of two given vectors. For clustering, OPTICS (Ankerst et al., 1999), an algorithm that is capable of outlier removal, is used with the default setting of scikit-learn (Pedregosa et al., 2011), except for setting xi to 0.03 and p to 1 to use noise robust L1 distance. In all processes, the random seed is set to 10. α, which determines the reflection ratio of the lexical and semantic scores, is set to 0.7. The minimum number of samples a cluster must

[1] https://huggingface.co/google/flan-t5-large

Extractor	Dataset (WER)						Time (s)	Δ Time (s)
	54		44		22			
	F1	EM	F1	EM	F1	EM		
BM25 (lexical)	46.51	**36.66**	49.29	37.63	52.31	39.95	**17.98**	-
RT-VQ^{2}A^2 (semantic)	35.19	26.60	33.20	24.93	37.27	27.88	18.79	**0.81**
BM25 + RT-VQ^{2}A^2 (dual)	**46.60**	36.58	**49.84**	**38.19**	**53.12**	**40.58**	19.38	1.4
DPR (semantic)	40.87	31.83	43.75	33.22	46.89	34.81	**40.48**	**22.50**
BM25 + DPR (dual)	**47.30**	**37.05**	**50.54**	**38.64**	**53.08**	**40.34**	40.76	22.78
Oracle Segment (upper bound)	61.56	48.28	62.59	48.36	67.63	52.10	-	-

Table 2: The evaluation result of QA on Spoken-SQuAD. **Bold** indicates the best performance in each separate environment separated by double line. Underline means the best performance among all methods, excluding the upper bound performance of the oracle segment.

Module	DPR	RT-VQ^{2}A^2
Window Extractor	0.94	1.37
Semantic Vector Extractor	29.12	**6.80**
Question Embedding Extractor	27.21	39.11
Semantic Matcher	4.24	0.47
Lexical Matcher	0.77	0.56
Answer Extractor (QA)	37.72	51.69
Total	100 (%)	100 (%)

Table 3: Percentage of time spent on each component in the entire pipeline when performing QA for DPR and RT-VQ^{2}A^2.

Extractor	Precision@1		
	WER54	WER44	WER22
BM25 (lexical)	0.630	0.661	0.657
RT-VQ^{2}A^2 (semantic)	0.449	0.404	0.439
Ours (dual)	**0.631**	**0.668**	**0.664**

Table 4: Window-level retrieval performance measured by precision@1 for WER 54, WER 44, and WER 22.

have is set to 14 for WER 54, 10 for WER 44, and 4 for WER 22.

5. Result

5.1. QA Performance Evaluation

In Table 2, we compare the performance of each extractor on questing answering task by F1 and EM (exact matching) metrics, using the Spoken-SQuAD dataset. Our proposed matching approach, leveraging both RT-VQ^{2}A^2, which utilizes quantized vectors obtained through a codebook, and BM25 as a lexical matching method, shows the best performance. Additionally, even when only the quantized semantic vector is utilized, it still produces some level of performance, indicating that the required information for QA has been appropriately captured. Furthermore, we include the results of the DPR approach, which directly extracts semantic vectors using LM for all document segments to compare performance.

Note that, as previously mentioned, the DPR approach is not practical for scenarios where the model processes requested audio documents and queries in real-time. This can be confirmed by the fact that it takes more than twice as long when using DPR together than lexical matching only. In contrast, our proposed method is more efficient without GPU support, as it shows little difference in processing time from the lexical matching with higher performance and is more than 10 times faster than DPR when excluding the time for lexical matching. In order to compare only the time taken to process the semantic matcher, we conduct the comparison based on the difference value obtained by subtracting the time taken to process the lexical matching method, involving the load of a model or preparing data, from each methodology. One noteworthy observation is that, when WER is 22, the proposed quantized semantic vector outperforms the use of DPR. This indicates that our proposed method is very fast, robust to error, and preserves semantic information well. In addition, as the WER increases, the performance gain decreases when using our proposed method. This is because it becomes difficult to filter out noise and group documents accurately when the WER increases.

5.2. Profiling the Execution Time

Table 3 shows the execution time of each module in running the SQA system with semantic vectors generated by DPR and RT-VQ^{2}A^2. The QA model accounts for the highest proportion of the overall execution time, as it utilizes a deep neural network that employs both an encoder and a decoder. The second most time-consuming modules are the semantic vector extractor and question embedding extractor, which use the encoder of the language model to extract embeddings with semantic information. In this step, we propose RT-VQ^{2}A^2, which utilizes a codebook to obtain a quantized semantic

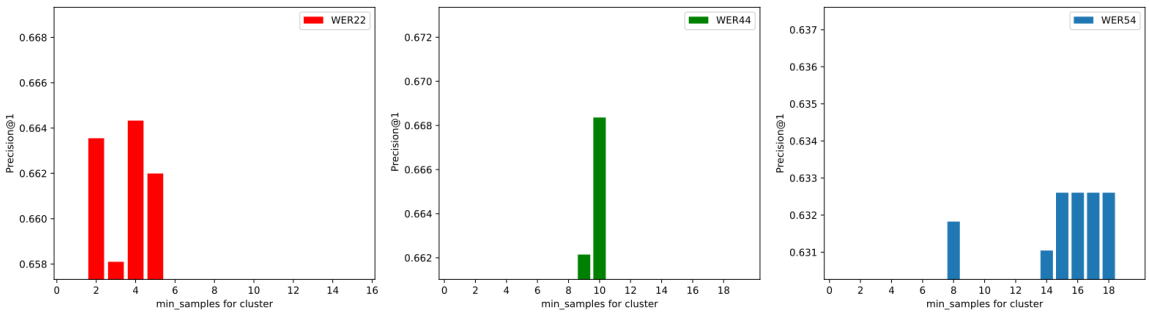

Figure 2: The graph showing the performance change of precision@1 on three different WERs, according to the minimum number of samples a cluster must have, only indicates cases where retrieval performance increased compared to BM25.

vector and reduces the proportion of time taken by the semantic vector extractor to a single-digit percentage of the overall execution time.

5.3. Retrieval Performance Evaluation

In Table 4, we report the window-level retrieval performance of different systems on three SQA dataset settings in terms of precision@1, measured as the percentage of top 1 retrieved window that contain the answer. The retrieval performance of the proposed method, using a quantized semantic vector with BM25, outperforms all WER settings, showing the clear advantage of using our, which can better handle the ASR error. Also, the performance improvement shows a similar tendency to that of the QA results.

6. Analysis

6.1. Relation between WER and Cluster

Figure 2 shows the results of performing three settings on how search performance changes based on precision according to the change in the minimum number of samples a cluster should have. The graph only indicates the cases where the performance increased compared to using BM25. As the WER increases , it can be observed that raising the minimum number of samples is necessary to obtain better retrieval performance. In other words, this result aligns with our intuition that clustering, where a large number of similar samples are grouped, can mitigate the impact of ASR errors when extracting semantic vectors. Additionally, the experimental result indicates that the number of clusters is crucial, as too many clusters cannot effectively remove ASR errors, while too few clusters can result in significant information loss. Based on these results, we utilize different minimum sample sizes for each WER in our experiments. The variation in the minimum samples for cluster results in each WER having a distinct cluster count, as shown in Table 1.

Clustering	# Cluster	F1	EM
K-means	32	**52.28**	**39.80**
K-means	64	51.95	39.80
K-means	512	51.83	39.49
K-means	1024	51.27	38.79
DBSCAN	971	52.23	39.64
OPTICS	1151	**53.12**	**40.58**

Table 5: The ablation study with different cluster algorithms and the number of cluster. Bold indicates the best performance for each cluster algorithm separated by double line. The underline represents the best performance among all methods when varying the clustering algorithms and the number of clusters.

6.2. Clustering Algorithm

To investigate the impact of cluster algorithm for codebook preparation, we perform an ablation study on WER 22 dataset. The setting, except for the clustering algorithm, is the same as in the all experiment. As we can see from Table 5, when we change the cluster algorithm to others, the performance drops in both the F1 and EM. In particular, when selecting a method vulnerable to outliers such as K-means (Sculley, 2010) cluster algorithm, the performance dramatically decreased. This means that filtering the text embedding outliers created by ASR error is necessary. Therefore, cluster algorithms capable of outlier removal, such as DBSCAN (Ester et al., 1996) and OPTICS, show better performance. These clustering algorithms have the advantage that users do not need to specify the number of clusters in advance, and do not need to have prior knowledge about the data being analyzed. However, a limitation of these algorithms is that when the text corpus used for clustering be-

comes too large, the number of resulting clusters can become excessively high.

6.3. The number of Cluster

In Table 5, we conduct the experiments by varying the number of clusters in the K-means clustering algorithm. The Table 5 represents the top two cases that achieve similar performance to OPTICS with significantly fewer clusters and the bottom two cases with a similar number of clusters to OPTICS but much worse performance. As mentioned in Section 6.2, OPTICS, which can remove outliers by itself, achieves much better results than K-means when both algorithms have the same number of clusters. Another noteworthy finding is that the performance of K-means clustering improves as the number of clusters decreases. This is because even though K-means cannot remove outliers, having a large number of samples in each cluster can reduce the influence of error. Furthermore, this discovery is consistent with the trends observed in Section 6.1 and Section 6.2.

7. Conclusion

In this work, we build an RT-VQ^{2}A^2 framework consisting of three steps: codebook preparation, quantized semantic vector extractor, and dual segment selector. The proposed RT-VQ^{2}A^2 is designed to perform real-time QA based on LLMs, within the audio document requested for ASR. In addition, we first pioneer the study of obtaining quantized semantic vectors at high speed through clustering and the pre-prepared codebook, rather than directly getting semantic vectors calculated through an LLM. Furthermore, we found that as ASR error increases, larger clusters consisting of more samples are needed to reduce the impact of noise. Moreover, we empirically demonstrate the importance of dealing with outliers in text corpus environments with ASR error. Our proposed model is validated on the Spoken-SQuAD dataset achieving higher scores across all environments rather than lexical matching only. Our framework achieves comparable QA performance compared to using authentic semantic vectors directly extracted from LLMs, with significantly better time efficiency and lower latency for inference.

8. Limitations

We shed the light on a method for extracting noise-agnostic semantic vectors for real-time QA on the text results of ASR. However, there are still some areas where this experiment can be improved. First, we conduct the research with the framework where QA is performed with ASR results on audio documents. It can be expanded to perform ASR on user queries as well as audio documents, which allow for completely audio-based QA. Second, the clustering method used to create the codebook can be improved by reflecting the intuition of the researcher about words that ASR does not do well, such as proper nouns. Finally, we empirically prove that clustering methods that can automatically exclude outliers are effective. However, there is a lack of semantic analysis of the information in outliers, such as what ASR errors mostly correspond to outliers. Our next research will focus on complementing these points.

9. Bibliographical References

Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Devi Parikh, and Dhruv Batra. 2015. Vqa: Visual question answering. *International Journal of Computer Vision*, 123:4–31.

Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. 1999. Optics: ordering points to identify the clustering structure. In *ACM SIGMOD Conference*.

Qingqing Cao. 2022. Mobivqa: Efficient on-device visual question answering. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 6:44:1–44:23.

Qingqing Cao, Noah Weber, Niranjan Balasubramanian, and Aruna Balasubramanian. 2019. Deqa: On-device question answering. *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*.

Liping Chen and Junchao Ren. 2020. Deep passage retrieval reranking based on semantic enhancement.

Zihang Chen, Hongbo Zhang, Xiaoji Zhang, and Leqi Zhao. 2017. Quora question pairs.

Chenyu, Nuo Chen, and Yuexian Zou. 2021. Mrd-net: Multi-modal residual knowledge distillation for spoken question answering. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3985–3991. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Eunsol Choi, Daniel Hewlett, Jakob Uszkoreit, Illia Polosukhin, Alexandre Lacoste, and Jonathan Berant. 2017. Coarse-to-fine question answering for long documents. In *Proceedings of the*

55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 209–220, Vancouver, Canada. Association for Computational Linguistics.

Yung-Sung Chuang, Chi-Liang Liu, and Hung yi Lee. 2019. Speechbert: Cross-modal pre-trained language model for end-to-end spoken question answering. *ArXiv*, abs/1910.11559.

Hyung Won Chung, Le Hou, S. Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Wei Yu, Vincent Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed Huai hsin Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *ArXiv*, abs/2210.11416.

J. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

Robert Dale. 2021. Gpt-3: What's it good for? *Natural Language Engineering*, 27:113 – 118.

Franck Dernoncourt, Trung Bui, and W. Chang. 2018. A framework for speech recognition benchmarking. In *INTERSPEECH*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Knowledge Discovery and Data Mining*.

Yang Gao, Zhengyu Pan, Honghao Wang, and Guanling Chen. 2018. Alexa, my love: Analyzing reviews of amazon echo. In *2018 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computing, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pages 372–380.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. Conformer: Convolution-augmented transformer for speech recognition. *ArXiv*, abs/2005.08100.

Yosuke Higuchi, Brian Yan, Siddhant Arora, Tetsuji Ogawa, Tetsunori Kobayashi, and Shinji Watanabe. 2022. Bert meets ctc: New formulation of end-to-end speech recognition with pre-trained masked language model. In *Conference on Empirical Methods in Natural Language Processing*.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*.

Kyungho Kim, Kyungjae Lee, Seung won Hwang, Young-In Song, and Seungwook Lee. 2021. Query generation for multimodal documents. In *Conference of the European Chapter of the Association for Computational Linguistics*.

Mika Koivisto and Simone Grassini. 2023. Best humans still outperform artificial intelligence in a creative divergent thinking task. *Scientific Reports*, 13.

Fangjun Kuang, Liyong Guo, Wei Kang, Long Lin, Mingshuang Luo, Zengwei Yao, and Daniel Povey. 2022. Pruned rnn-t for fast, memory-efficient asr training. In *Interspeech*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc V. Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Veton Këpuska and Gamal Bohouta. 2018. Next-generation of virtual personal assistants (microsoft cortana, apple siri, amazon alexa and google home). In *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 99–103.

Chia-Hsuan Lee, Yun-Nung (Vivian) Chen, and Hung yi Lee. 2019. Mitigating the impact of speech recognition errors on spoken question answering by adversarial domain adaptation. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7300–7304.

Seungyoung Lim Myungji Kim Jooyoul Lee. 2018. Korquad: Korean qa dataset for machine comprehension.

Jie Lei, Licheng Yu, Mohit Bansal, and Tamara L. Berg. 2018. Tvqa: Localized, compositional video question answering. In *EMNLP*.

Patrick Lewis, Barlas Oğuz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2020. Mlqa: Evaluating cross-lingual extractive question answering. *ArXiv*, abs/1910.07475.

Qing Li, Qingyi Tao, Shafiq R. Joty, Jianfei Cai, and Jiebo Luo. 2018. Vqa-e: Explaining, elaborating, and enhancing your answers for visual questions. *ArXiv*, abs/1803.07464.

Seungyoung Lim, Myungji Kim, and Jooyoul Lee. 2019. Korquad1.0: Korean qa dataset for machine reading comprehension. *ArXiv*, abs/1909.07005.

Guan-Ting Lin, Yung-Sung Chuang, Ho-Lam Chung, Shu wen Yang, Hsuan-Jui Chen, Shuyan Dong, Shang-Wen Li, Abdelrahman Mohamed, Hung yi Lee, and Lin shan Lee. 2022a. Dual: Discrete spoken unit adaptive learning for textless spoken question answering.

Guan-Ting Lin, Yung-Sung Chuang, Ho-Lam Chung, Shu wen Yang, Hsuan-Jui Chen, Shuyan Dong, Shang-Wen Li, Abdel rahman Mohamed, Hung yi Lee, and Lin-Shan Lee. 2022b. Dual: Discrete spoken unit adaptive learning for textless spoken question answering. In *Interspeech*.

Zhaojiang Lin, Bing Liu, Andrea Madotto, Seungwhan Moon, Paul A. Crook, Zhenpeng Zhou, Zhiguang Wang, Zhou Yu, Eunjoon Cho, Rajen Subba, and Pascale Fung. 2021. Zero-shot dialogue state tracking via cross-task transfer. *ArXiv*, abs/2109.04655.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*.

Jirí Martínek, Christophe Cerisara, Pavel Král, Ladislav Lenc, and Josef Baloun. 2022. Weak supervision for question type detection with large language models. In *Interspeech*.

Merve Ünlü Menevşe, Ebru Arisoy, and Murat Saraçlar. 2019. Question answering for spoken lecture processing. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7365–7369.

Maryam Mirzaei, Kourosh Meshgi, and Tatsuya Kawahara. 2017. Detecting listening difficulty for second language learners using automatic speech recognition errors. pages 156–160.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. 2022. Training language models to follow instructions with human feedback. *ArXiv*, abs/2203.02155.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

Denis Peskov, Joe Barrow, Pedro Rodriguez, Graham Neubig, and Jordan Boyd-Graber. 2019. Mitigating Noisy Inputs for Question Answering. In *Proc. Interspeech 2019*, pages 789–793.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *ICML*.

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *ACL*.

Abhilasha Ravichander, Siddharth Dalmia, Maria Ryskina, Florian Metze, Eduard Hovy, and Alan W Black. 2021a. Noiseqa: Challenge set evaluation for user-centric question answering.

Abhilasha Ravichander, Siddharth Dalmia, Maria Ryskina, Florian Metze, Eduard H. Hovy, and Alan W. Black. 2021b. Noiseqa: Challenge set evaluation for user-centric question answering. *ArXiv*, abs/2102.08345.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3:333–389.

Erich Schubert and Michael Gertz. 2018. Improving the cluster structure extracted from optics plots. In *Lernen, Wissen, Daten, Analysen*.

D. Sculley. 2010. Web-scale k-means clustering. In *The Web Conference*.

Chenglei Si, Ziqing Yang, Yiming Cui, Wentao Ma, Ting Liu, and Shijin Wang. 2021. Benchmarking robustness of machine reading comprehension models. *ArXiv*, abs/2004.14004.

Georgios Sidiropoulos, Svitlana Vakulenko, and Evangelos Kanoulas. 2022. On the impact of speech recognition errors in passage retrieval for spoken question answering. In *Proceedings of the 31st ACM International Conference on Information &amp Knowledge Management*. ACM.

Dan Su and Pascale Fung. 2020. Improving spoken question answering using contextualized word representation. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8004–8008.

Hao Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *ArXiv*, abs/1908.07490.

Zhen-Quan Tang, Benyou Wang, and Ting Yao. 2022. Dptdr: Deep prompt tuning for dense passage retrieval. In *International Conference on Computational Linguistics*.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. In *Rep4NLP@ACL*.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2021a. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021b. mt5: A massively multilingual pre-trained text-to-text transformer. In *NAACL*.

Yi Yang, Wen tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *EMNLP*.

Zekun Yang, Noa García, Chenhui Chu, Mayu Otani, Yuta Nakashima, and Haruo Takemura. 2020. Bert representations for video question answering. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1545–1554.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*.

Chenyu You, Nuo Chen, Fenglin Liu, Shen Ge, Xian Wu, and Yuexian Zou. 2022. End-to-end spoken conversational question answering: Task, dataset and model. In *NAACL-HLT*.

Chenyu You, Nuo Chen, and Yuexian Zou. 2020. Knowledge distillation for improved accuracy in spoken question answering. *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7793–7797.

10. Language Resource References

Chia-Hsuan Lee, Szu-Lin Wu, Chi-Liang Liu, and Hung yi Lee. 2018. Spoken squad: A study of mitigating the impact of speech recognition errors on listening comprehension. In *Interspeech*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *EMNLP*.