CMUQA in QALab-3: Essay Question Answering for University Entrance Exams

Fadi Botros
Carnegie Mellon University
fbotros@andrew.cmu.edu

Francesco Ciannella
Carnegie Mellon University
fciannella@cmu.edu

Takaaki Matsumoto
Carnegie Mellon University
tmatsumo@andrew.cmu.edu

Evan Chan
Carnegie Mellon University
yiksanc@andrew.cmu.edu

Cheng-Ta Chung
Carnegie Mellon University
chengtac@cs.cmu.edu

Lucas Bengtson
Carnegie Mellon University
lbengtso@cs.cmu.edu

Keyang Xu
Carnegie Mellon University
keyangx@andrew.cmu.edu

Tian Tian
Carnegie Mellon University
tian.tian@cmu.edu

Teruko Mitamura
Carnegie Mellon University
teruko@cs.cmu.edu

ABSTRACT

This paper describes the CMUQA entry into NTCIR QA Lab-3. CMUQA is an essay-question answering system that uses Wikipedia as its knowledge base. The essay-question portion of QA Lab-3 is composed of real exam questions from the University of Tokyo's entrance exams which are focused on world history. The proposed system formulates answers to these questions using the following sequence of processes: question analysis, document retrieval, sentence extraction, sentence scoring, text ordering and text summarization. Despite the difficulty of this challenge and the small size of training data, CMUQA achieved the highest expert score amongst the competing end-to-end systems in the task.

Keywords

Question answering, open knowledge base, summarization, NTCIR-13, world history

Team Name

CMUQA

Subtasks

English subtask (Phase 2: End-to-End for Essay Questions)

1. INTRODUCTION

Question answering (QA) is one of the most notable natural language processing applications today and has been heavily researched for several decades. While most research focuses on factoid, true/false and multiple choice QA tasks, essay QA has been proven to be one of the more challenging tasks since it usually requires a deeper understanding of the subject matter, information extraction from multiple sources and summarization to produce a coherent essay.

NTCIR (NII Testbeds and Community for Information access Research) [1] is a series of workshops that expand research in Information Access (IA) technologies including information retrieval, question answering, text summarization, extraction, etc. QA Lab [2], one of the tasks of NTCIR, aims to investigate complex real-world QA technologies as a joint effort of participants. The QA tasks of the NTCIR 13 QA Lab consist of three type of questions: multiple choice, named-entity and essay type questions from Japanese university entrance examination, which focus on world history [14][12][13].

In this paper, we present our system which participated in the essay QA portion of QA Lab 3. The rest of the paper is laid out as follows. We further explain the task in section 2. In section 3, we discuss the design of the system in detail and show evaluation for each module. In section 4 we present end-to-end system evaluation. We finally finish off with a discussion and conclusion.

2. TASK DESCRIPTION

The essay QA task of NTCIR QA Lab 3 contains short/simple and long/complex essays. The former requires an one or two sentence long answer (from 15 to 60 words) with some questions resembling factoid type questions. The latter expects a 5-8 sentence long answer (approx. 225-270 words) and the question includes a list of 5-8 keywords that should be used in the essay. Examples of the questions are following:

Short essay The Inca Empire had no writing system, but it controlled the large territory of the Andes. Describe, in 15 English words, the transportation and information methods used by the Empire.

Long essay In answer space (A), in 225 English words or less, describe the historical significance of the philosophies of these intellectuals, including the conditions in the 18th century which led them to these conclusions, especially in France and China. Use each of the terms below once, and underline each term when it is used; Society of Jesus, imperial examinations, enlightenment, absolute monarchy, revocation of the Edict of Nantes, French Revolution, class system, Literary Inquisition.

3. SYSTEM DESCRIPTION

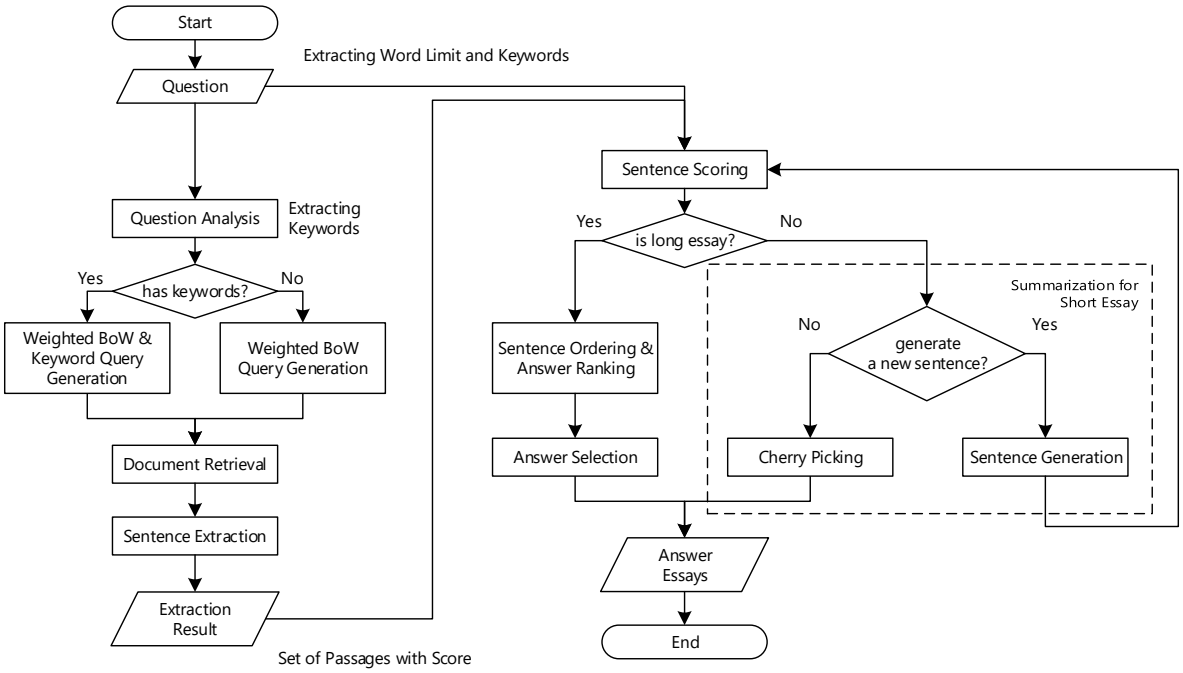

Figure 1: CMUQA System Flowchart

3.1 Overview

Fig. 1 shows the architecture of the end-to-end system. Detailed architecture, algorithms used and experimental design for each of these modules are covered in detail in the following subsections. Communication between each module is performed using JSON files for ease of use and readability.

The main data source used to extract answers is the history section of Wikipedia. [7].

3.2 Question Analysis

The question analysis module is meant to extract the useful information from the question text to be used by the rest of the system. This module is composed of three components, namely, information extraction, text processing and weighted keywords generation.

The module first extracts all data from the question file which is an XML format. It extracts the different parts of the question, along with the expected answer lengths.

In the text processing phase, the system removes redundant text from the questions. For example, sentences like "Write your answers in the answer space" are removed since they are not useful to the system.

The following task is to generate keywords from the question that can be used by the document retrieval module. It does so by simply extracting the terms in the question with the highest tf-idf score. It calculates tf-idf from all the text in the questions and Wikipedia. The terms along with their tf-idf weights are then passed on to the following modules.

3.3 Document Retrieval

The document retrieval module indexes information from Wikipedia and retrieves relevant text records using structured queries generated from the questions. The Wikipedia history subset created by Wang et al. [16] was used as the collection for constructing the index. Indri [15] was utilized for indexing, which was stopped using the default Indri stoplist and stemmed using the Krovetz stemmer.

Each Wikipedia page can be indexed as a document, which is the basic unit for retrieval. This is known as the page-level indexing. However, the question answer requires to locate the exact paragraph in extracting specific sentences relevant to the question and the whole Wikipedia page might contain too much noise. In order to increase the accuracy of sentence extraction, we adopted passage-level indexing; it divided each Wikipedia page into sentences using Standford CoreNLP tool [1] and used a sliding window approach to combine sentences into passages [4]. Here, the sliding windows contains 10 sentences without overlapping and we heuristically chose 10 because the question answer is required to have around 40-60 words.

Structured queries are generated with the weighted keywords extracted from the Question Analysis module(see Section 3.2). An example for keywords set "Olympia; Greek; 4th century CE " is shown as follows:

$$\#combine(\alpha_1\ Olympia\ \alpha_3\ Greek\ \alpha_3\ \#1(4th\ century\ CE))$$

where α_1,α_2,α_3 are weights generated by the Question Anal-

[1] https://stanfordnlp.github.io/CoreNLP/

ysis module; And #1() operator requires all terms inside to appear continuously.

The retrieval model is Indri [11], which combines statistical language models and Bayesian inference networks. All parameters were default settings. Top 20 retrieved documents are ranked with Indri scores and returned for the Sentence Extraction module in the next step.

3.4 Sentence Extraction

This module takes the output of the question analysis and document retrieval modules and extracts sentences that could be potential answers to the question. Since short and long essay questions have different answering requirements, the system uses different strategies to answer them. This module consists of following sub-modules:

Document Cleaning Raw Wikipedia files are highly noisy: they contain a lot of tables, links, citations, markup, etc. That is why it is important to clean the documents and remove all the unnecessary content. This sub-module also segments the documents into sentences and tokenizes each sentence. These are the steps that are taken to process each document:

1. Remove documents that do not actually contain any useful text but rather contain a list of links to other pages (e.g. Category pages)
2. Segment documents into sentences
3. Filter out sentences that:
   (a) Contain links
   (b) Contain HTML or Wikipedia markup
   (c) Are image captions
4. Tokenize each sentence:
   (a) Remove non-alphanumeric characters
   (b) Remove stopwords
   (c) Lowercase tokens
   (d) Stem tokens
5. Remove sentences that contain two tokens or less
6. Remove duplicate sentences

Passage Extraction There are separate passage extraction sub-modules for short and long essays since different strategies are used to answer each type of question. Each sub-module takes in the output of the question analyzer, the cleaned documents and outputs a list of sentences that are potential answers to the question. The algorithms used by these sub-modules are outlined in detail in the following section.

3.4.1 Algorithms

The following algorithms were tested to select the most suitable algorithms for sentence extraction.

Jaccard Similarity Similarity is calculated between all the words in the question (introduction/instruction paragraphs and given keywords) and each sentence from the retrieved documents then the top 10 sentences with the highest scores are chosen.

Field-weighted Jaccard Similarity : Since the introduction paragraph is usually longer than the instruction paragraph, the sentences extracted using Jaccard similarity tended to be more relevant to the introduction paragraph but not to the actual instruction paragraph. Therefore, to remedy this problem, the following formula was used to give more weight to the instruction paragraph:

$$\begin{aligned}\text{Score}(\text{Question}, \text{Sentence}) = \\ 0.7 * \text{Jaccard}(\text{Instruction}, \text{Sentence}) \\ + 0.3 * \text{Jaccard}(\text{Introduction}, \text{Sentence})\end{aligned} \quad (1)$$

Field-weighted Jaccard + MMR Wikipedia contains many sentences that are very similar to each other in terms of content. Therefore, sometimes the system would return 10 sentences that are all very similar. This is not very beneficial for this task, especially for long essay questions where we want to cover a wide range of topics. To diversify the extracted sentences, MMR is used. The essence of MMR [5], which is a greedy algorithm, is in each iteration, it would pick passage that has high relevant score with question but also with little overlap with the already selected passages.

Field-weighted TF-IDF History questions tend to contain many names of people, events, places and special words that should be given more weight since the question is usually focused on those words. Therefore, TF-IDF and cosine similarity are used to rank sentences.

Field-weighted TF-IDF + PM2 Long essay questions contain keywords that have to be used and discussed in the essay. However, the previous methods cannot guarantee that all keywords were covered in the extracted passages. It is possible that all the extracted passages are only relevant to one keyword (or none at all). The PM2 diversification algorithm [6] is used to try to increase keyword coverage for long essay questions. PM2 is generally used in document retrieval when the query can have multiple intents and we want to retrieve documents that address all the intents proportionally. It gives a higher score to documents that cover multiple intents. Similarity, in long essay questions we want to retrieve sentences that cover each keyword proportionally and give extra weight to sentences that cover multiple keywords. Sentences that cover multiple keywords can connect the given concepts and potentially produce a more coherent essay.

3.4.2 Evaluation

We focused on human annotations to evaluate the extracted passages. A binary relevance metric was used to evaluate each extracted passage and precision @10 and mean reciprocal rank were then calculated for each experiment. For long essay questions, keyword recall is also evaluated by measuring the fraction of keywords that are present in the extracted passages.

Table 1 summarizes the results for each of the tested algorithms. The results show that field-weighted TF-IDF + PM2 gave the best results for all metrics.

As mentioned above, using simple Jaccard similarity is naive since most of the extracted passages were relevant to the introduction paragraph but not to the actual question. Using field-weighted Jaccard doubled the precision scores

which indicates that it is effective. While MMR reduced redundancy, the results show that it didn't improve precision or MRR thus it is questionable whether it is useful or not for this task. As predicted, TF-IDF was a very effective method to improve results as demonstrated by the improved precision and MRR. However, TF-IDF gave the lowest keyword recall for long essays but PM2 proved effective as it doubled keyword recall while also improving precision/MRR for long essays.

Table 1: Evaluation Result of Passage Extraction Algorithms

Algorithm	Short Essays		Long Essays		
	P@10	MRR	P@10	MRR	Keyword Recall
Jaccard	0.077	0.330	0.520	0.850	0.447
Field-weighted Jaccard	0.150	0.432	0.700	0.767	0.509
Field-weighted Jaccard + MMR	0.109	0.444	0.660	0.733	0.529
Field-weighted TF-IDF	**0.191**	**0.447**	0.679	0.750	0.376
Field-weighted TF-IDF + PM2	-	-	**0.720**	**0.900**	**0.714**

3.5 Sentence Scoring

The purpose of the sentence scoring module is to give scores to the extracted passages from long essay-questions. The scores represent how well each passage covers the topics of the given keywords. The simplest sentence scoring method is measuring keyword density in a passage:

$$\text{Score} = \frac{k_s}{m} \tag{2}$$

where k_s is the number of keywords in the sentence, and m is the number of words in the sentence. All keywords and words from the extracted passages are stemmed. Stop words and punctuations are removed before calculation.

However, a passage may cover a keyword topic very well but still not have an exact match with that keyword. This reformulation takes advantage of word vectors to calculate similarity between words in a passage and the given keywords:

$$\text{Score} = \sum_{i=1}^{m} \frac{\max(w_i \cdot k_1, w_i \cdot k_2, ...w_i \cdot k_n)}{\log m} \tag{3}$$

where, m is the number of words in the sentence, n is the number of keywords, w is the vector for word i in an extracted passage, t is the keyword vector. GloVe (6B 100d) [10] word embeddings were used.

3.6 Text Ordering for Long Essay

The purpose of the text ordering module is to sort the extracted passages to create an essay that is coherent, that covers all the necessary topics and that is below the required answer length.

3.6.1 K-Means model

In [17], Zhang proposed summary generation by using global and local coherence. Global coherence refers to connectivity between remote sentences and sub-topic transitions in texts. For example, in an essay, usually the "cause" of a historical event is introduced first, then the "result" of that event is presented. On the other hand, local coherence indicates the connectivity between adjacent sentences, such as using some transition words to connect two sentences. Because coherence can be regarded as some kind of similarity between sentences, thus, this module adopts cosine similarity to measure the coherence.

To achieve coherence, the K-means algorithm in scikit-learn package [3] is used to cluster the input passages. It is assumed that each cluster is related to different sub-topics, as the similarity within each sub-topic should be very similar. Each passage is represented as a word vector, whose value is TF-IDF of the words in each dimension.

After clustering, the next step is to generate the order of these clusters, the sequence of sub-topics, to achieve global coherence. To do this, the system first picks the cluster that is the most similar to the other clusters, then it greedily picks clusters that is the most similar to the already selected clusters. To achieve local coherence, the same method is used to order individual passages in a cluster.

3.6.2 Evaluation

To identify the optimal number of clusters for the task, we tested the method using various K values. The dataset used was the gold standard passages and essays provided by NTCIR. There are 5 long essay questions and each of them is associated with several passages and 3 gold standard essays. In this evaluation, the gold standard passages are used as input to the system, and gold standard essays are used to evaluate the essay generation system. The evaluation metrics are ROUGE-1 and ROUGE-2 means[8].

Table 2 shows the performance for with the model with different K values. We can see that ROUGE-1 score is the same for all K-Means methods, it is because the sentence removal strategy would remove almost the same sentences, and ROUGE-1 only measures one single words. For ROUGE-2, the score is different as it measures bi-grams. It improves when K grows from 1 to 3, then decreases gradually after that. It indicates that the clustering is effective, while the number of clusters should not be too large, as there are generally around 7 to 9 sentences in the gold standard essays.

Table 2: Performance of the K-means model for sentence ordering using various K parameters

K	ROUGE-1	ROUGE-2
1	0.584	0.356
3	0.584	0.358
5	0.584	0.357
7	0.584	0.352

3.7 Summarization for Short Essay

Short essay questions usually require complex answers that discuss topics from several sources and that are under a given word limit in length. The summarization for short essay module provides a way to summarize a set of sentences coming from the upper layers to produce a fixed length short essay, following the directions provided in the question.

The module uses two possible summarization techniques. The first technique is an abstractive summarization technique using AMR [9] which attempts to summarize a list of passages that were extracted from earlier modules. The second technique simply returns the best passage from the extracted passages using some simple rules.

3.7.1 AMR model

Abstractive summarization is one of the hard NLP tasks that is still an open field of research with very few techniques, unlike other NLP tasks. It is a task that cannot be decoupled from semantics: to be able to create an abstract summary from a set of passages, one needs to have a deep insight into what is the meaning that the passages bear. Therefore we thought to use Abstractive Meaning Representation (AMR) [9], which is one of the resources available in NLP for implementing semantics. A thorough description of the algorithms can be found in [9]. We implemented the algorithms described in that paper, except for the generative model which isn't discussed throroughly in the paper. The generative model is the part that creates a well formed sentence from an AMR graph. For now, our method outputs a bag of words out of the AMR graph.

AMR provides a whole-sentence semantic representation, represented as a rooted, directed, acyclic graph. Nodes of an AMR graph are labeled with concepts, and edges are labeled with relations. Concepts can be English words, PropBank event predicates, or special keywords. The core semantic relations are adopted from the PropBank annotations; other semantic relations include "location," "mode," "name," "time," and "topic."

In the AMR summarization framework, summarization consists of three steps

1. parsing the input sentences to individual AMR graphs,
2. combining and transforming those graphs into a single summary AMR graph
3. generating text from the summary graph.

The graph summarizer, first merges AMR graphs for each input sentence through a concept merging step, in which coreferent nodes of the graphs are merged; a sentence conjunction step, which connects the root of each sentence's AMR graph to a dummy "ROOT" node; and an optional graph expansion step, where additional edges are added to create a fully dense graph on the sentence level. These steps result in a single connected source graph. A subset of the nodes and arcs from the source graph are then selected for inclusion in the summary graph. Ideally this is a condensed representation of the most salient semantic content from the source.

We used the proxy report section of the AMR Bank because a dataset for a summarization task should include inputs and their summaries, each with gold-standard AMR annotations. A proxy report is created by annotators based on a single newswire article, selected from the English Gigaword corpus.

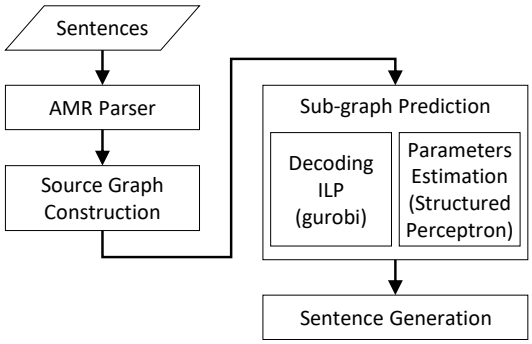

Figure 2: AMR Model Architecture

Fig. 2 shows the architecture of the AMR model. Brief description of each component are presented:

Source Graph Construction The "source graph" is a single graph constructed using the individual sentences' AMR graphs by merging identical concepts. Concept merging involves collapsing certain graph fragments into a single concept, then merging all concepts that have the same label.
Ideally, a source graph should cover all of the gold standard edges, so that summarization can be accomplished by selecting a subgraph of the source graph

Subgraph Prediction This step selects a summary subgraph from the source graph. This is done with a structured prediction algorithm that enforces the following constraints in the statistical model for subgraph selection: include information without altering the meaning, maintain brevity, and produce fluent language. The selection of the graphs is done using ILP (Integer Linear Programming)

Decoding Decoding is performed as an ILP task with the constraints that the output forms a connected subcomponent of the source graph.
The length constraint is used to fix the size of the summary graph (measured by the number of edges). This is an important parameter in that the performance of a summarization system depends strongly on their compression rate, and it is important for this task because of the length limitations on the essays. An exact ILP solver called Gurobi is used.

Parameter Estimation Source graphs and summary graphs, represent a collection of input and output pairs, therefore we can use a Machine Learning algorithm like the structured perceptron to learn the parameters of the objective function designed in the previous set.

Generation Generation is the weakest link in the current chain. At the moment it is no more than a bag of words, but the plan is to plug it into a language generator from AMR.

3.7.2 Pick one sentence

The pick one sentence method simply selects one sentence among the candidate ones, based on the relevance score provided by the scoring system and by the closeness to the required sentence's length.

It uses simple methods to clip sentences at punctuation marks or by prunning the lexical parse tree in order to fit a sentence under the word limit.

4. END-TO-END EVALUATION

Table 3 shows the different configurations of the CMUQA system that were submitted to the NTCIR QA Lab-3 task. Table 4 presents some of the results of CMUQA and the competing systems on the test dataset [13]. CMUQA, Forst, IMTKU and MTMT are all end-to-end systems that performed both extraction of passages and summarization, while the DGLab system only did summarization given a pool of all the passage extractions from the other systems. Therefore, these systems can not be directly compared.

Automated evaluation was performed on both short and long essay answers, and human evaluation was performed on long essay answers only. ROUGE-1 and 2 were used for automated evaluation. A standard rubric was used for human evaluation that graded each long essay based on the number of grammatical errors, semantic errors and missing keywords. According to this rubric, grammatical errors include unusual spelling, extra and/or unnecessary words and phrases. Semantic errors include sentences which cannot be understood in the context or that contain false facts. The number of missing keywords (KW) is the number of given keywords that were not included in the answer.

According to the expert scores, CMUQA outperformed all the other end-to-end systems. While all system answers were scored as zero by the expert, we still can evaluate answer qualities by allowing the scores to take negative values. CMUQA lost most of its point from grammatical errors but it had the lowest number of missing keywords. It performed competitively in most of the ROUGE metrics, expect in short essay ROUGE scores, where it has almost doubled the score of most of the other systems. This might indicate that generative summarization is important for this task. The results of CMUQA1 and CMUQA2 show that sentence scoring increased performance a bit but by a small margin.

Qualitative analysis of the answers show that most of the grammatical errors were because of unnecessary Wikipedia content that was not properly removed from the data. This includes lists, markup language, document titles, foreign language, etc. While generative summarization achieved high ROUGE scores, most of the answers given by the AMR model were highly ungrammatical since as mentioned in section 3.7, the method did not have a generative model to produce well formed sentences, it simply outputs a bag of words.

4.1 Example Answers

The system answers and gold standards for the example questions shown in Section 2 are following:

Gold Standard for Short Essay
It used roads around Cuzco and knotted ropes called quipu.

System Answer (Pick One) for Short Essay
Inca road system.

System Answer (Generative) for Short Essay
road developed system

Gold Standard for Long Essay
The Society of Jesus, which engaged in missionary work overseas, was also active in China, bringing information about China to Europe. The scientific revolution of 18th century Europe brought about the Enlightenment, especially in France, with its focus on reason and equality. Voltaire praised China for lacking doctrines which were contrary to reason. This was in response to Catholic control of France since the reign of Louis XIV, who abolished the Edict of Nantes, which granted Protestant the same rights as Catholics. Reynal praised China for not having hereditary nobility. His aim was to contrast France, with its fixed class system, to China, whose appointment of ministers under the imperial examination system ensured some degree of social mobility. Montesquieu, however, criticized China's tyrannical authoritarian system. By criticizing China's restriction of free speech through the Literary Inquisition, he meant to implicitly criticize France's system of absolute monarchy. In these ways, the Enlightenment criticized France's authoritarian religion, class system, and absolute monarchy, and created the philosophical foundation of the French Revolution which overturned the absolute monarchy.

System Answer for Long Essay
For de Tocqueville, the Revolution was the inevitable result of the radical opposition created in the 18th century between the monarchy and the men of letters of the Enlightenment. It was instead the French Revolution, by destroying the old cultural and economic restraints of patronage and corporatism (guilds), that opened French society to female participation, particularly in the literary sphere.All this is not to say that intellectual interpretations no longer exist. By the end of the 18th century, prominent French philosophers and literary personalities of the day, including Anne-Robert-Jacques Turgot, were making persuasive arguments to promote religious tolerance. The edict paved the way for the most far-reaching reforms in terms of their social consequences, including the creation of a national education system and the abolition of the imperial examinations in 1905.

5. DISCUSSION

Even though CMUQA achieved the highest expert scores, it is important to note that all systems still received negative scores. Quantitative and qualitative analysis both make it clear that none of the systems are close to human performance. The questions were relatively difficult compared to standard factoid questions, since they usually required a deep understanding of the content matter and a high level of inference to give a proper answer.

Short essay questions proved to be harder to answer than long essay questions based on ROUGE scores and answer quality. Most of these answers required data from multiple sources, therefore simple retrieval methods performed poorly. This could be one reason why generative summarization achieved high ROUGE scores. However, current generative summarization models tend to give highly un-

Table 3: CMUQA system configurations

System Name	Scoring Method	Short Essay Summarization Method
CMUQA1	Extraction Score	Pick one sentence
CMUQA2	Word Similarity (Eq. 3)	Pick one sentence
CMUQA3	Word Similarity (Eq. 3)	Generative (AMR)

Table 4: NTCIR QA Lab-3 results for short and long essay questions [13]

System	Short Essay		Long Essay					
	ROUGE-1 Mean (Stopword)	ROUGE-2 Mean (Stopword)	ROUGE-1 Mean (Stopword)	ROUGE-2 Mean (Stopword)	Expert Score Mean	Num. of Grammatical Err. Mean	Num. of Semantic Err. Mean	Num. of Missing KW Mean
CMUQA1	0.0152	0.0054	0.1204	0.0115	0 (**-26.2**)	7.8	0.2	**3.6**
CMUQA2	0.0173	0.0054	0.1241	0.0119				
CMUQA3	**0.0442**	**0.0131**	0.1263	0.0119				
Forst1	0.0225	0.0027	0.0000	0.0000				
IMTKU1	0.0262	0.0000	0.1315	0.0069	0 (-29)	**6.0**	**0**	4.6
IMTKU2	0.0124	0.0000	0.0132	0.0004				
MTMT1	0.0370	0.0023	**0.1692**	**0.0190**	0 (-36.4)	10.8	1.8	4.4
MTMT2	0.0255	0.0000	0.1387	0.0107				
DGLab1 SumExP			0.1428	0.0139	0 (-22)	6.8	0.4	3.0
DGLab2 SumExP			0.1458	0.0138				

grammatical answers and the topic still remains a difficult NLP problem.

Given that most of the expert score points were lost due to grammatical errors, it is evident that data quality and proper data cleaning is highly important when it comes to generating essays. While we spent a lot of time cleaning the Wikipedia data, we were unable to clean it entirely since there were too many edge cases. The noise in the data tended to confuse the parsers and tokenizers in our system which resulted in poorly formed sentences in the output essays.

Another factor that rendered this task difficult, is that there were a couple questions that referred to accompanying content (most likely images or maps) that was simply not given with the questions. These questions are almost impossible to answer, even for humans. The systems gave highly varying answers for these questions.

However, one reason why CMUQA got the highest expert scores is because it had the lowest number of missing keywords amongst the end-to-end systems. This is an indicator that the diversification methods in the sentence extraction module were effective. Generally, in an essay, it is important to connect concepts together instead of having sentences that discuss topics independently. PM2 [6] proved to be an ideal method for this purpose.

The task of essay question answering is a difficult task in itself, however, the task of automated essay scoring is also a challenging problem. It is arguable whether ROUGE is a good metric for this task considering that it does not seem to be consistent or correlated with the expert scores as seen in table 4. The ideal scoring model would take grammar and semantics into consideration instead of simple exact word matching. Yet, another important note is that the training and testing data were very small, therefore it is difficult to reliably judge the performance of different methods and systems.

6. CONCLUSION

In this paper, we presented CMUQA, an essay question answering system that uses Wikipedia to answer questions from Tokyo University entrance exams. Six modules were described and tested, which include question analysis, document retrieval, sentence extraction, sentence scoring, sentence ordering and short essay generation. The proposed system extracts keywords from the question text and each keyword is given a weight using tf-idf. Relevant articles are retrieved from Wikipedia and important passages are extracted based on the weighted keywords. For long essays, the extracted passages are ordered to produce a coherent essay. For short essays, summarization techniques are used to compress the extracted passages to a short answer.

Task results demonstrated that this task is more challenging than standard QA tasks such as factoid and multiple-choice questions. It also showed that short essay questions are more challenging to answer than long essay questions. However, CMUQA achieved the highest expert score amongst all the end-to-end systems. Error analysis hinted that the diversification and summarization methods used were some of the reasons why CMUQA outperformed the other systems, however, more data would be needed to get more reliable results.

Acknowledgement

We thank Prof. Eric Nyberg for assistance with the development of the essay QA system.

7. REFERENCES

[1] *The 13th NTCIR*, 2017. http://research.nii.ac.jp/ntcir/ntcir-13/.

[2] *NTCIR-13 QA Lab-3*, 2017. http://research.nii.ac.jp/qalab/.

[3] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, et al. Api design for machine learning software: experiences from the scikit-learn project. *arXiv preprint arXiv:1309.0238*, 2013.

[4] J. P. Callan. Passage-level evidence in document retrieval. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 302–310. Springer-Verlag New York, Inc., 1994.

[5] J. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336. ACM, 1998.

[6] V. Dang and W. B. Croft. Diversity by proportionality: an election-based approach to search result diversification. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 65–74. ACM, 2012.

[7] D. Dua, B. Juneja, S. Agarwal, K. Sakamoto, D. Wang, and T. Mitamura. CMUQA: multiple-choice question answering at NTCIR-12 QA lab-2 task. In *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies, National Center of Sciences, Tokyo, Japan, June 7-10, 2016*, 2016.

[8] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8. Barcelona, Spain, 2004.

[9] F. Liu, J. Flanigan, S. Thomson, N. M. Sadeh, and N. A. Smith. Toward abstractive summarization using semantic representations. In R. Mihalcea, J. Y. Chai, and A. Sarkar, editors, *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 1077–1086. The Association for Computational Linguistics, 2015.

[10] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[11] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281. ACM, 1998.

[12] H. Shibuki, K. Sakamoto, M. Ishioroshi, A. Fujita, Y. Kano, T. Mitamura, T. Mori, and N. Kando. Overview of the NTCIR-12 QA lab-2 task. In *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies, National Center of Sciences, Tokyo, Japan, June 7-10, 2016*, 2016.

[13] H. Shibuki, K. Sakamoto, M. Ishioroshi, Y. Kano, T. Mitamura, T. Mori, and N. Kando. Overview of the NTCIR-13 QA lab-3 task. In *Proceedings of the 13th NTCIR Conference on Evaluation of Information Access Technologies, National Center of Sciences, Tokyo, Japan, December 5-8, 2017*, 2017.

[14] H. Shibuki, K. Sakamoto, Y. Kano, T. Mitamura, M. Ishioroshi, K. Y. Itakura, D. Wang, T. Mori, and N. Kando. Overview of the NTCIR-11 qa-lab task. In *Proceedings of the 11th NTCIR Conference on Evaluation of Information Access Technologies, NTCIR-11, National Center of Sciences, Tokyo, Japan, December 9-12, 2014*, 2014.

[15] T. Strohman, D. Metzler, H. Turtle, and W. B. Croft. Indri: A language model-based search engine for complex queries. In *Proceedings of the International Conference on Intelligent Analysis*, volume 2, pages 2–6. Amherst, MA, USA, 2005.

[16] D. Wang, L. Boytsov, J. Araki, A. Patel, J. Gee, Z. Liu, E. Nyberg, and T. Mitamura. CMU multiple-choice question answering system at NTCIR-11 qa-lab. In *Proceedings of the 11th NTCIR Conference on Evaluation of Information Access Technologies, NTCIR-11, National Center of Sciences, Tokyo, Japan, December 9-12, 2014*, 2014.

[17] R. Zhang. Sentence ordering driven by local and global coherence for summary generation. In *Proceedings of the ACL 2011 Student Session*, pages 6–11. Association for Computational Linguistics, 2011.