Extraction Meets Abstraction: Ideal Answer Generation for Biomedical Questions

Yutong Li[*] **Nicholas Gekakis**[*] **Qiuze Wu**[*] **Boyue Li**[*]
Khyathi Raghavi Chandu, Eric Nyberg
Language Technologies Institute, Carnegie Mellon University
{anareshk,hkesavam,madhurad,pkalwad,kchandu,teruko,ehn}@cs.cmu.edu

Abstract

The growing number of biomedical publications is a challenge for human researchers, who invest considerable effort to search for relevant documents and pinpointed answers. Biomedical Question Answering can automatically generate answers for a user's topic or question, significantly reducing the effort required to locate the most relevant information in a large document corpus. Extractive summarization techniques, which concatenate the most relevant text units drawn from multiple documents, perform well on automatic evaluation metrics like ROUGE, but score poorly on human readability, due to the presence of redundant text and grammatical errors in the answer. This work moves toward abstractive summarization, which attempts to distill and present the meaning of the original text in a more coherent way. We incorporate a sentence fusion approach, based on Integer Linear Programming, along with three novel approaches for sentence ordering, in an attempt to improve the human readability of ideal answers. Using an open framework for configuration space exploration (BOOM), we tested over 2000 unique system configurations in order to identify the best-performing combinations for the sixth edition of Phase B of the BioASQ challenge.

1 Introduction

Human researchers invest considerable effort when searching very large text corpora for answers to their questions. Existing search engines like PubMed (Falagas et al., 2008) only partially address this need, since they return relevant documents but do not provide a direct answer for the user's question. The process of filtering and combine information from relevant documents to obtain an ideal answer is still time consuming (Tsatsaronis et al., 2015). Biomedical Question Answering (BQA) systems can automatically generate ideal answers for a user's question, significantly reducing the effort required to locate the most relevant information in a large corpus.

Our goal is to build an effective BQA system to generate coherent, query-oriented, non-redundant, human-readable summaries for biomedical questions. Our approach is based on an extractive BQA system (Chandu et al., 2017) which performed well on automatic metrics (ROUGE) in the 5th edition of the BioASQ challenge. However, owing to the extractive nature of this system, it suffers from problems in human readability and coherence. In particular, extractive summaries which concatenate the most relevant text units from multiple documents are often incoherent to the reader, especially when the answer sentences jump back and forth between topics. Although the existing extractive approach explicitly attempts to reduce redundancy at the sentence level (via SoftMMR), stitching together existing sentences always admits the possibility of redundant text at the phrase level. We improve upon the baseline extractive system in 3 ways: (1) re-ordering the sentences that are selected by the extractive algorithm; (2) fusing words and sentences to form a more humanireadable summary; and (3) using automatic methods to explore a much larger space of system configurations and hyperparameter values when optimizing system performance. We hypothesize that the first two techniques will improve the coherence and human readability, while the third technique provides an efficient framework for tuning these approaches in order to maximize automatic evaluation (ROUGE) scores.

2 Overview of Baseline System Architecture

In this section, we provide a brief layout of our baseline system, which achieved the top ROUGE scores in the final test batches of the fifth edition of BioASQ Challenge (Chandu et al., 2017). This system includes baseline modules for relevance ranking, sentence selection, and sentence tiling.

The baseline relevance ranker performs the fol-

[*] denotes equal contribution

lowing steps: 1) Expand concepts in the original question using a metathesaurus, such as UMLS (Bodenreider, 2004) or SNOMEDCT (Donnelly, 2006); and 2) calculate a relevance score (e.g. Jaccard similarity) for each question/snippet pair (to measure relevance) and each pair of generated snippets (to measure redundancy). The baseline sentence selection model used the Maximal Marginal Relevance (MMR) algorithm (Carbonell and Goldstein, 1998), which iteratively selects answer sentences according to their relevance to the question and their similarity to sentences that have already been selected, until a certain number of sentences have been selected. The baseline sentence tiling module simply concatenates selected sentences up to a given limit on text length (200 words), with no attempt to module or improve the coherence of the resulting summary.

The baseline system achieved high ROUGE scores, but performed poorly on the human readability evaluation in BioASQ 2017. In order to improve human readability, we first developed several post-processing modules, such as sentence reordering and sentence fusion, which will be discussed in detail in following sections.

3 Sentence Ordering

3.1 Motivation

As discussed in Section 1, we tried to improve upon the Soft MMR system (Chandu et al., 2017). This pipeline assumes the relevance to be a proxy for ordering the selected sentences to generate the final summary. On the other hand, it does not take into account the flow and transition of sentences to build a coherent flow between these sentences. Since the maximum length of the answer is 200 words (as imposed by the guidelines of the competition), this system optimizes on selecting the most non-redundant query relevant sentences to maximize the ROUGE score. In this section, we focus on different types of sentence ordering that lead to more coherent answers.

3.2 Algorithms and Techniques

3.2.1 Similarity Ordering

The intuition behind the Similarity Ordering algorithm is that sentences that have similar content should appear consecutively so that the generated answer is not jumping back and forth between topics. Our implementation is based on work by Zhang (2011), which discusses the use of similarity metrics at two levels - first to cluster sentences, and then to order them within a cluster - which can lead to big improvements in coherency and readability. We apply this approach to the BQA domain, where we cluster our set of candidate answers using k-means with $k = 2$. We then order the sentences within each cluster, starting with the candidate sentence nearest to the centroid of its cluster and working outward. The intuition is that the most central sentence will contain the largest number of tokens shared by all the sentences in the cluster, and is therefore likely to be the most general or comprehensive sentence in the cluster. This supports our goal of an ideal answer that begins with a broad answer to the question, followed by specifics and supporting evidence from the literature.

In Figure 1a we see that the order of the sentences that appear in the final answer is completely independent of their ordering in the original snippets.

3.2.2 Majority Ordering

The Majority Ordering algorithm Barzilay and Elhadad (2002) makes two main assumptions that are quite reasonable: sentences coming from the same parent document should be grouped together, and the most coherent ordering of a group of sentences is how they were presented in their parent document. Topically, it is logical that sentences drawn from the same parent document would be similar. Grammatically and syntactically, it is logical that the sentences may be structured in a way such that maintaining an invariant ordering would augment human comprehension.

Specifically, the Majority Ordering algorithm groups sentences by their parent document and then orders the blocks by the ranking of the highest ranked sentence in a block. Figure 1 illustrates the differences between Similarity Ordering, Majority Ordering, and Block Ordering. The color of each sentence unit indicates the document it was selected from, and the suffix indicates the relevance score of that unit within the document.

3.2.3 Block Ordering

Intuitively, the Block Ordering algorithm is an amalgamation of the Similarity Ordering and Majority Ordering algorithms. The Block Ordering algorithm has two primary components. The first component involves grouping the sentences into blocks based on their parent document. This step is shared between the Block Ordering algorithm and the Majority Ordering algorithm. The second step involves ordering the grouped blocks of text.

The algorithm for ordering the blocks of texts combines document heuristics with our Similarity Ordering algorithm. We first order the blocks by their length (the number of sentences in teh block). For blocks of equal length, we calculate the similarity of each block with the last fixed sentence. Hence, given the last sentence of the preceding block, we select the next block first by its length, and then by the similarity of the block with the preceding sentence. If there is no single longest block to begin the answer, then we select the longest block that is most similar to the entire answer. This algorithm is tuned for specific goals with respect to human comprehension and readability. Grouping the sentences into blocks is done to maximize local coherence. The use of block length as an ordering heuristic is done to order topics by relevance. Finally, ordering blocks of equal length by similarity to the preceding sentence is done to maximize sentence continuity and fluidity.

In Figure 1c the green block is ordered first because it is the longest. The blue block is ordered second because it has the highest similarity score with sentence 3.4. The yellow block is ordered third because it has a higher similarity with sentence 2.2, and the red block is thus last.

3.3 Quantitative Analysis

To evaluate our approaches, we performed a manual analysis of 100 different answers, ordered by each of our proposed ordering algorithms (see Table 1). We rate each ordering as 'reasonable' or 'unreasonable'. Note that this rating does not pass judgment on the correctness of the answer, since it is designed for a comparative analysis at the module level (i.e. to compare ordering approaches rather than content selection).

Algorithm	**Reasonable**	**Unreasonable**
Baseline	59	41
Similarity Ordering	55	45
Majority Ordering	71	29
Block Ordering	75	25

Table 1: Manual evaluation of sentence ordering

3.4 Qualitative Analysis

Because sentence ordering in the baseline system is based solely on question-answer relevance, we identified two major issues: global coherence and local coherence.

The global coherence issue is generally a problem of layout and cohesiveness. An ideal answer would begin with a broad answer to the question and move into more specific details and any available evidence to support the answer. Further, an ideal answer should not be hopping back and forth between topics and should stick to one before moving on to another. The baseline system did a decent job of beginning with a broad answer to the question because the input sequence is ordered by their relevance score. However after the first sentence, answers tended towards redundant information and divergent trains of thought.

The local coherence issue has more to do with the semantics of the sentence and grammatical restrictions of the language. For instance, language like 'There was also' should not appear as the first sentence in an answer because this makes no sense logically. Additionally certain words like 'Furthermore' indicate that the content of the sentence is highly dependent on the content of the preceding sentence(s), and this dependency is frequently broken by the baseline ordering approach.

3.4.1 Similarity Ordering

We found that the Similarity Ordering performed poorly; only 55 of 100 answers were deemed 'reasonable'. We believe that this is due to the high degree of similarity between the candidate sentences in our domain. Because the candidate sentences are so similar to each other, the results of clustering are highly variant and appeared to be almost arbitrary at times. All the sentences contain similar language and key phrases that makes it difficult to create meaningful sub-clusters. Additionally, one of the biggest problems with our system is due to the sentences that began with phrases like 'However' and 'Furthermore' that place strict requirements on the content of the preceding sentence. This was particularly problematic for the Similarity Ordering algorithm which has no mechanism for making sure that such sentences are placed logically with their dependent sentences. The Similarity Ordering algorithm does perform relatively well in creating logical groups of sentences that cut down on how often an answer is jumping from one topic to another. Additionally these groups are ordered well, beginning with the more general of the two and then finishing with specifics and a presentation of the supporting data. However, we note that the problems with local coherence greatly outweigh the strengths in global coherence since a good answer can still be coherent, even if the organization could be improved, whereas if local coherence is poor, then the answer becomes nonsensical.

3.4.2 Majority Ordering

The Majority Ordering algorithm proved to be a successful method for ordering sentences, where 71 out of 100 answers were deemed 'reasonable'. The Majority Ordering displayed very strong local coherence, which confirms the hypothesis that sentences should likely be kept in their original ordering to maximize human readability and coherence.

However, this algorithm faced issues with global coherence. It produced answers that start

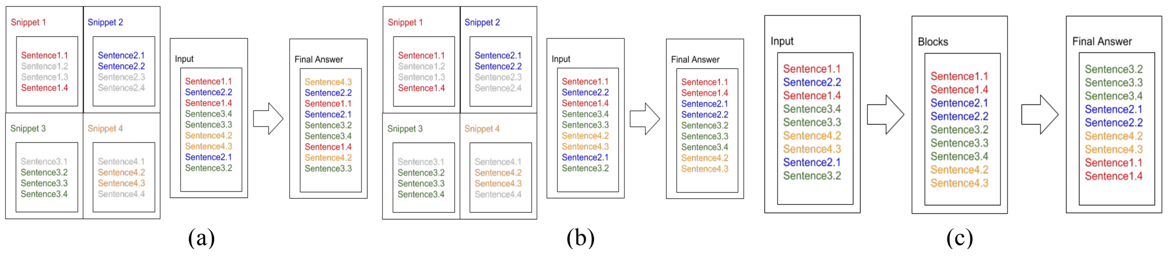

Figure 1: (a) Similarity Ordering (b) Majority Ordering (c) Block Ordering

with a relevant topic more often than not; however, after the initial block, it struggled to smoothly transition from one block to the next. This is consistent with expectations for the Majority Ordering algorithm. The block with highest rated sentence is ordered first, which explains why the first block is frequently the most topically relevant. After the initial block placement, however, the algorithm makes no explicit attempts to manage or smooth transitions between blocks. Compared with the other two algorithms, this is where the Majority Ordering algorithm displays its poorest performance. It performs strongly when ordering sentences within a block, enforcing local coherence so that sentences beginning with language such as ‘Finally’, ‘Lastly’, ‘Therefore’, etc. followed a related sentence that satisfied the sequential dependency.

3.4.3 Block Ordering

The Block Ordering algorithm produced the best answers, with 75 out of 100 answers ranked as ‘reasonable’. This is consistent with our expectations, as the Block Ordering algorithm effectively combines the strongest aspects of the Majority Ordering and Similarity Ordering algorithms. With respect to local coherence, this algorithm displays similar performance when compared to the Majority Ordering algorithm, while displaying stronger coherence between blocks (due to the use of a similarity metric to order blocks). This algorithm also displayed the strongest global coherence, which is likely due to first grouping the sentences into blocks and then ordering them.

This algorithm displayed one core weakness, which is its inability to identify high-quality opening sentences. This is due to the usage of block length as a heuristic for topic relevance. While in the majority of cases this heuristic proved to be successful, accounting for these outliers may significantly improve the performance of the Block Ordering algorithm. We note that the Block ordering algorithm performed well in producing high-quality, coherent answers; although the development of coherence models and measures is not the main focus of this paper, we can see that Block Ordering performs the best with respect to the simple coherence evaluation we conducted.

4 Sentence Fusion

An observed weakness of the original system is that the generated summaries often contain highly repetitive information. While MMR is added in the pipeline to deal with redundancy and maximize the diversity of covered information, extractive summarization still picks entire sentences that may partially overlap with a previously selected sentence. To tackle this problem, we introduce sentence fusion as a way to identify common information among sentences and apply simple abstractive techniques over the baseline extractive summaries.

4.1 Methodology

Given a set of candidate sentences generated by the pipeline for each summary, the sentence fusion module operates in two steps: 1) the candidate set is expanded to include fused sentences, and 2) sentences are selected from the expanded set to produce a new summary.

4.1.1 Expansion of Candidate Set

To generate fused sentences, we begin by building upon previous work on multiple-sentence compression (Filippova, 2010), in which a directed word graph is used to express sentence structures. The word graph is constructed by iteratively adding candidate sentences. All words in the first sentence are added to the graph by creating a sequence of word nodes. A word in the following sentence is then mapped onto an existing word node if and only if it is the same word, with the same part of speech. Our assumption is that a shared node in the word graph is likely to refer to the same entity or event across sentences.

We then find a K-possible fused sentence by searching for the K-shortest path within the word graph. Definition of the edge weights follows from the original paper (Filippova, 2010):

$$w(e_{ij}) = \frac{\frac{freq(i)+freq(j)}{\sum_{s\in S} diff(s,i,j)^{-1}}}{freq(i) \times freq(j)}$$

where $diff(s, i, j)$ is the difference between the offset positions of word i and j in sentence s. Intuitively, we want to promote a connection between two word nodes with close distance, and between nodes that have multiple paths between them. We also prefer a compression path that goes through the most frequent no-stop nodes to emphasize important words.

When applying the sentence fusion technique to the BioASQ task, we first pre-process the candidate sentences to remove transition words like 'Therefore' and 'Finally'. Such transition words may be problematic because they are not necessarily suitable for the new logical intent in fused sentences, and may break the coherence of the final answer. We also constrain fusion so that the fused sentences are more readable. For instance, we only allow fusing of pairs of sentences that are of proper length, in order to avoid generating overly complicated sentences. We also avoid fusing sentences that are too similar or too dissimilar. In the first case, information in the two sentences is largely repetitive, so we simply discard the one containing less information. In the latter case, fusing two dissimilar sentences more likely confuses the reader with too much information rather than improving the sentence readability. Finally, we add a filter to discard ill-formed sentences, according to some hand-crafted heuristics.

4.1.2 Selecting Sentences from Candidate Set

The next step is to select sentences from the candidate set and produce a new summary. An Integer Linear Program (ILP) problem is formulated as follows, according to (Gillick and Favre, 2009):

$$\max_{y,z} \sum_{i=1}^{N} w_i z_i, \text{ such that } \sum_{j=1}^{M} A_{ij} y_j \geq z_i, A_{ij} y_j \leq z_i,$$
$$\sum_{j=1}^{M} l_j y_j \leq L, y_j \in \{0,1\}, z_i \in \{0,1\}$$

In the equation, z_i is an indicator of whether concept i is selected into the final summary, and w_i is the corresponding weight for the concept. The goal is to maximize the coverage of important concepts in a summary. During the actual experiments, we assign diminishing weights so that later occurrences of an existing concept are less important. This forces the system to select a more diverse set of concepts. We follow the convention of using bigrams as a surrogate for concepts (Taylor Berg-Kirkpatrick and Klein, 2011; Dan Gillick and Hakkani-Tur, 2008), and bigram counts as initial weights. Variable A_{ij} indicates whether concept i appears in sentence j, and variable y_j indicates if a sentence j is selected or not.

4.2 Discussion

Table 2 shows the results of different configurations of the ordering and fusion algorithms (Rows 1 - 4, Row 7, Row 9). Though the overall ROUGE score drops slightly from 0.69 to 0.61 after sentence fusion with the ILP-selection step, this is still competitive with other systems (including the baseline). The sentence re-ordering does not directly impact the ROUGE scores.

We manually examined the fused sentences for 50 questions. We found that our sentence fusion technique is capable of breaking down long sentences into independent pieces, and is therefore able to disregard irrelevant information. For example, given a summary containing the original sentence:

'Thus, miR-155 contributes to Th17 cell function by suppressing the inhibitory effects of Jarid2. (2014) bring microRNAs and chromatin together by showing how activation-induced miR-155 targets the chromatin protein Jarid2 to regulate proinflammatory cytokine production in T helper 17 cells.'

our fusion technique is able to extract important information and formulate it into complete sentences, producing a new summary containing the following sentence:

'Mir-155 targets the chromatin protein jarid2 to regulate proinflammatory cytokine expression in th17 cells.'

The fusion module is also able to compress multiple sentences into one, with minor grammatical errors. For example:

Sentence 1: *'The RESID Database is a comprehensive collection of annotations and structures for protein post-translational modifications including N-terminal, C-terminal and peptide chain cross-link modifications[1].'*

Sentence 2: *'The RESID Database contains supplemental information on post-translational modifications for the standardized annotations appearing in the PIR-International Protein Sequence Database[2]'*

our approach produces the fused sentence:

'The RESID Database contains supplemental information on post-translational modifications[1] is a comprehensive collection of annotations and structures for protein post-translational modifications including N-terminal, C-terminal and peptide chain cross-link modifications[2].'

However, the overall quality of fused sentences is not stable. As shown in Figure 2, around 25% of the selected sentences in final summaries are

	Method	Rouge-2	Rouge-SU4	Avg Precision	Avg Recall	Avg F1	Avg Length
1	**Baseline System**	0.6948	0.6890	0.2297	0.8688	0.3207	173.31
2	**MMR + Order**	0.6291	0.6197	0.2758	0.8118	0.3633	140.39
3	**MMR + Fusion**	0.6183	0.6169	0.2783	0.8094	0.3687	139.24
4	**MMR + Relevance + Order**	0.6357	0.6256	0.2728	0.8124	0.3606	143.55
5	**MMR + Relevance + Order + Post**	**0.6215**	**0.6126**	**0.2788**	**0.8111**	**0.3668**	**139.10**
6	**MMR + Relevance + Order + Fusion + LM**	0.6114	0.6042	0.2775	0.8113	0.3682	141.21
7	**MMR + Relevance + Order + Fusion**	0.6213	0.6101	0.2686	0.8099	0.3579	143.94
8	**MMR + Relevance + Order + Fusion + Post**	0.6017	0.5932	0.2775	0.8091	0.3653	140.38
9	**MMR + Fusion + Order**	0.6223	0.6159	0.2840	0.8181	0.3745	138.79
10	**MMR + Fusion + Relevance + Order**	**0.6257**	**0.6214**	**0.2825**	**0.8193**	**0.3730**	**139.73**
11	**MMR + Fusion + Relevance + Order + Post**	0.6149	0.6096	0.2886	0.8126	0.3768	136.43
12	**Fusion + MMR + Relevance + Order**	0.6112	0.6103	0.2837	0.8211	0.3723	142.11
13	**Fusion + MMR + Relevance + Order + Post**	0.6048	0.6040	0.2898	0.8143	0.3789	137.78

Table 2: Performance of different module combinations on Test Batch 4, BioASQ 4th edition.

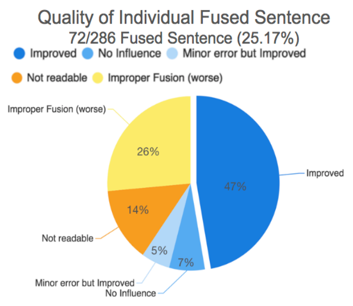

Figure 2: Quality of Fused Sentences

fused. Among the fused sentences, 47% improved the overall readability by reducing redundancy and repetition. 5% of the sentences have improved readability with minor grammatical errors, such as a missing subordinate conjunction or superfluous discourse markers. 8% of the fused sentences did have an appreciable effect on readability. However, a large number of fused sentences (around 26 %) were not coherent and degraded the quality of the answer.

5 Further Improvements

In order to further improve the performance of our system, we made a few modifications to each module in the system, and improved the overall architecture of the module pipeline:

• **Modification of System Architecture:** We intuited that the ILP process in the sentence fusion model could not handle a very large number of candidate inputs, producing a lot of (redundant, similar) fused sentences. In order to resolve this problem, we removed the ILP model from the sentence fusion step, and moved the sentence fusion step before the sentence selection module (Rows 12-13), so that the MMR algorithm in the sentence selection module could take care of eliminating redundant fused sentences.

• **Modifications to Sentence Selection Module and Relevance Ranker:** For the sentence selection module, we modified the original MMR model. The original MMR model selected a fixed number of sentences, which naturally introduced repetition. In order to reduce repetition, we built a so called '*Early-Stop MMR*' which stops selecting sentences when maximum overlap score grows beyond a certain threshold and minimum relevance score drops down below another threshold (Rows 4-8).

For the relevance ranker, we explore an alternative similarity metric ((Row 6). The Query Likelihood Language Model (Schütze et al., 2008) is widely used in information retrieval. We formulated the relevance ranking procedure as an information retrieval problem and used a language model, so that long sentences would get higher penalty.

• **Post-Processing:** To further reduce repetition, we add an additional filter before final concatenation by iteratively adding the selected sentences to the final output, and discarding a sentence if it is too similar to the existing summary (Rows 8,11 and 13se).

6 Configuration Space Exploration

Configuration Space Exploration (CSE) is the technique of trying different combinations of configurations of all modules to find the best configuration (Yang et al., 2013; Yang, 2016). We used the BOOM framework to explore and optimize the space of hyperparameters and module configurations. We explored 2,268 unique configurations of three different hyperparameters: α, used for the MMR module; k, used for the clustering-based Ordering module; and a token limit, used in the Tiling module. Figure 3 shows the pipeline structure we used.

• **Alpha:** This parameter of the MMR module

	Question	Which syndrome is associated with mutant DVL1?
	Ideal Answer	**Mutations in DVL1 cause an osteosclerotic form of Robinow syndrome**.
1	**MMR + Relevance + Order + Fusion**	We identified de novo frameshift mutations in DVL1, a mediator of both canonical and non-canonical Wnt signaling, as the cause of RS-OS, an RS subtype involving osteosclerosis, in three unrelated individuals. Argeted Sanger sequencing in additional subjects with DRS uncovered DVL1 exon 14 mutations in five individuals, including a pair of monozygotic twins. DVL1 frameshift mutations clustering in the penultimate exon cause autosomal-dominant Robinow syndrome. **Mutations in DVL1 cause an osteosclerotic form of Robinow syndrome**.
2	**MMR + Relevance + Fusion + Order**	**Mutations in DVL1 cause an osteosclerotic form of Robinow syndrome**. DVL1 frameshift mutations clustering in the penultimate exon cause autosomal-dominant Robinow syndrome. We identified de novo frameshift mutations in DVL1, a mediator of both canonical and non-canonical Wnt signaling, as the cause of RS-OS, an RS subtype involving osteosclerosis, in three unrelated individuals. Argeted Sanger sequencing in additional subjects with DRS uncovered DVL1 exon 14 mutations in five individuals, including a pair of monozygotic twins.
3	**Fusion + MMR + Relevance + Order + Post**	**DVL1 frameshift mutations in DVL1 cause an osteosclerotic form of Robinow syndrome.** We identified de novo frameshift mutations in DVL1, a mediator of both canonical and non-canonical Wnt signaling, as the cause of RS-OS, an RS subtype involving osteosclerosis, in three unrelated individuals. Argeted Sanger sequencing in additional subjects with DRS uncovered DVL1 exon 14 mutations in five individuals, including a pair of monozygotic twins.

Table 3: System performance comparing Fusion + Ordering and Ordering + Fusion

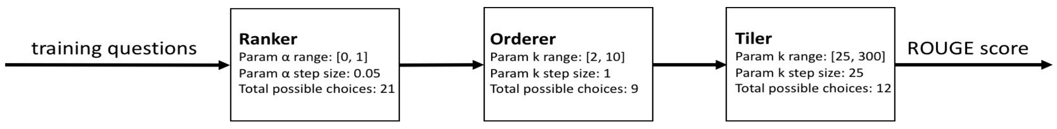

Figure 3: Structure of the CSE pipeline.

controls the trade-off between snippet similarity to the question and snippet similarity to already selected snippets. In our experiments alpha was varied between 0 and 1 at intervals of .05. We have found that the ideal value for alpha is 0.1.

- **Number of Clusters:** The k in the Ordering module controls the number of clusters used to order the snippets for the clustering-based Sentence Ordering algorithms. A small k value produces few, general clusters, while a large k value produces many highly specific clusters with the danger of creating clusters that are actually meaningless or having many clusters that contain a single sentence. In our experiments, k was tested at values from 2 to 10. Although the effect on Rouge score was very small, we have found that the ideal value for k is 3. A caveat to this result is that we are measuring the effect hyperparameter k has on the final Rouge scores achieved by the system. Since the purpose of k is to assist in sentence ordering, not precision or recall, we would expect that adjusting k would have a negligible impact on the Rouge score. Further parameter tuning is needed in cases like this where the primary effect of the parameter is not easily captured by Rouge.

- **Token Limit:** The token limit is used by the Tiling module to set a maximum number of allowed tokens in the answer. If the cumulative token count of the selected snippets exceeds the token limit then sentences will be removed from the end of the final answer until the token limit is satisfied. In our experiments the token limit was tested at values from 25 to 300 in increments of 25. We have found that the ideal value for the token limit is 100.

The two distinct clusters found in the histogram shown in Figure 4 are entirely explained by the token limit. All scores less than 0.27 were obtained by configurations where the token limit was set to 25. The rest of the scores, all above 0.28 were obtained by configurations where the token limit was greater than or equal to 50. In addition to the Rouge score penalty for extremely low token limits, we observed a significant, though much smaller, penalty for token limits of 150 and greater.

7 Results

7.1 Analysis: Effects of Individual Modules

Table 2 shows the results of extensions to the baseline system. Two systems are highlighted (Rows 5 and 10)), as they give the most balanced results between the quality of retrieved information and conciseness: one system performs sentence selection, then ranks sentences prior to ordering by relevance, and applies the additional post-processing step (Row 5); the other system performs sentence

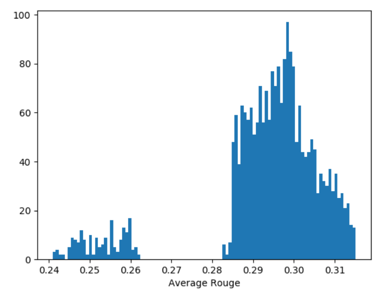

Figure 4: Distribution of Rouge scores across all hyperparameter configurations.

selection, fusion, and then ranks sentences prior to ordering without the post-processing step (Row 10).

Rows 5, 8, 11 and 13 show the effectiveness of the additional post-processing step. Overall, this procedure is able to reduce the answer length, while preserving important information. We observed that the post-processing step is less effective when fusion is performed after MMR. This is because in these settings, there is an additional sentence selection step in the fusion module using integer linear programming that forces the selected sentences to be diverse. In all other settings, including when fusion is performed prior to MMR, we only have one sentence selection step. Since MMR iteratively selects sentences according to both similarity and relevance, the last selected ones may be informative but repetitive. Row 6 shows our experiments with language modeling; the language model gives a higher penalty to longer sentences, which produces shorter but less informative results.

7.2 Analysis: Impact of System Architecture

Exploring the performance of systems using different architectures, we observed that systems with fusion prior to ordering can generate more logically coherent summaries. Table 3 shows an example. All underlined sentences express the same fact that DVL1 is the cause of Robinow syndrome. In Row 1, where fusion is performed after ordering, there is a sentence that serves like an explanation between the underlined sentences, which breaks the logical coherence. In Row 2 and Row 3 where ordering is performed after fusion, the generated answers demonstrate better coherence: All underlined sentences are placed together, following by the explanation; The opening sentences are also more concise and more directly related to the question.

We also experimented with architectures where the fusion module is run prior to MMR, and MMR is used as the only sentence selection step. In these systems, MMR receives many fused sentences that overlap and complement each other at the same time, because all similar sentences are fused prior to sentence selection. As a result, such architectures sometimes produce summaries that are more repetitive compared to others.

8 Conclusion and Future Work

Though extractive summarization techniques can be developed to maximize performance as measured by evaluation metrics like ROUGE, such systems suffer from human readability issues as mentioned above. In this paper we attempted to combine extractive techniques with simple abstractive extensions, by extracting the most relevant non-redundant sentences, re-ordering and fusing them to make the resulting text more human-readable and coherent. Using an initial set of 100 candidate answer sets, we experimented with different ordering algorithms such as Similarity, Majority and Block Ordering, and identified that Block Ordering performs better the others in terms of global and local coherence. We then introduced an Integer Linear Programming based fusion module that is capable of not only fusing repeated content, but also breaks down complicated sentences into simpler sentences, thus improving human readability. The improved baseline system achieved a ROUGE-2 of 0.6257 and ROUGE-SU4 of 0.6214 on test batch 4 of BioASQ 4b. We acknowledge that providing immediate human feedback during the BioASQ competition is manually expensive, although this would greatly help in tuning our systems. We were able to perform a manual evaluation on a sub-sample of the data, in order to introduce the use of human evaluation during system development. We also incorporated an automatic evaluation framewook (BOOM) which allowed us to test many different system configurations and hyperparameter values during system development. As BOOM is completely general and can be applied to any pipeline of Python modules, this adaptation was relatively straightforward, and allowed us to automatically test more than 2,000 different system configurations.

In the future, we would like to explore parameter tuning for sentence ordering using human evaluation metrics. There are several additional refinements (abstractions) of the extracted sentences which rely on simple post-processing or text cleaning methods which could be performed before sentences are passed to the fusion module. Another interesting direction that we would explore is the possibility of automatically predicting reasonable sentence orderings.

References

Regina Barzilay and Noemie Elhadad. 2002. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17:35–55.

Olivier Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270.

Jaime Carbonell and Jade Goldstein. 1998. 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.

Khyathi Chandu, Aakanksha Naik, Aditya Chandrasekar, Zi Yang, Niloy Gupta, and Eric Nyberg. 2017. Tackling biomedical text summarization: Oaqa at bioasq 5b. *BioNLP 2017*, pages 58–66.

Benoit Favre Dan Gillick and Dilek Hakkani-Tur. 2008. The icsi summarization system at tac 2008. In *In Proceedings of TAC*.

Kevin Donnelly. 2006. Snomed-ct: The advanced terminology and coding system for ehealth. *Studies in health technology and informatics*, 121:279.

Matthew E Falagas, Eleni I Pitsouni, George A Malietzis, and Georgios Pappas. 2008. Comparison of pubmed, scopus, web of science, and google scholar: strengths and weaknesses. *The FASEB journal*, 22(2):338–342.

Katja Filippova. 2010. Multi-sentence compression: Finding shortest paths in word graphs. In *In Proceedings of the 23rd International Conference on Computational Linguistics*, pages 322–330. ACL.

Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *In Proceedings of NAACL*.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*, volume 39. Cambridge University Press.

Dan Gillick Taylor Berg-Kirkpatrick and Dan Klein. 2011. Jointly learning to extract and compress. In *In Proceedings of ACL*.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, et al. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, 16(1):138.

Zi Yang. 2016. *Analytics Meta Learning*. Ph.D. thesis, University of Illinois at Urbana-Champaign.

Zi Yang, Elmer Garduno, Yan Fang, Avner Maiberg, Collin McCormack, and Eric Nyberg. 2013. Building optimal information systems automatically: Configuration space exploration for biomedical information systems. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 1421–1430. ACM.

Renxian Zhang. 2011. 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.