Data Augmentation for Abstractive Query-Focused Multi-Document Summarization

Ramakanth Pasunuru,[1] Asli Celikyilmaz,[2] Michel Galley,[2] Chenyan Xiong,[2] Yizhe Zhang,[2] Mohit Bansal,[1] Jianfeng Gao[2]

[1]UNC Chapel Hill, [2]Microsoft Research, Redmond

{ram, mbansal}@cs.unc.edu, {aslicel, mgalley, Chenyan.Xiong, yizhe.zhang, jfgao}@microsoft.com

Abstract

The progress in Query-focused Multi-Document Summarization (QMDS) has been limited by the lack of sufficient large-scale high-quality training datasets. We present two QMDS training datasets, which we construct using two data augmentation methods: (1) transferring the commonly used single-document CNN/Daily Mail summarization dataset to create the QMDSCNN dataset, and (2) mining search-query logs to create the QMDSIR dataset. These two datasets have complementary properties, i.e., QMDSCNN has real summaries but queries are simulated, while QMDSIR has real queries but simulated summaries. To cover both these real summary and query aspects, we build abstractive end-to-end neural network models on the combined datasets that yield new state-of-the-art transfer results on DUC datasets. We also introduce new hierarchical encoders that enable a more efficient encoding of the query together with multiple documents. Empirical results demonstrate that our data augmentation and encoding methods outperform baseline models on automatic metrics, as well as on human evaluations along multiple attributes.

1 Introduction

Query-focused multi-document summarization (QMDS) aims at generating a short summary from a set of documents that answers a query. Compared to the popular single document summarization (SDS) task (Rush, Chopra, and Weston 2015; Chopra, Auli, and Rush 2016; Nallapati et al. 2016; Celikyilmaz et al. 2018; Chen and Bansal 2018; Gehrmann, Deng, and Rush 2018), research in QMDS has received less attention. This is partially due to the scarcity of large-scale high-quality QMDS datasets. The SDS has variety of high-quality datasets (Hermann et al. 2015; Grusky, Naaman, and Artzi 2018) on different domains such as news (Napoles, Gormley, and Durme 2012; Nallapati et al. 2016), scientific articles (Qazvinian and Radev 2008), etc., however, not many high-quality datasets exist for QMDS training and evaluation.

Another overlooked feature of QMDS is that it tries to solve more realistic query-based scenarios than the SDS or multi-document summarization (MDS) tasks. Different from these tasks, QMDS task considers summarizing only salient information that best answers the query in a logical order. In fact, QMDS is more realistic for various applications such as personalized information retrieval (IR), conversational IR, and recommendation engines, in which search results can be tailored to an information need. To support research on challenging QMDS task, we introduce two data augmentation methods and new neural models.

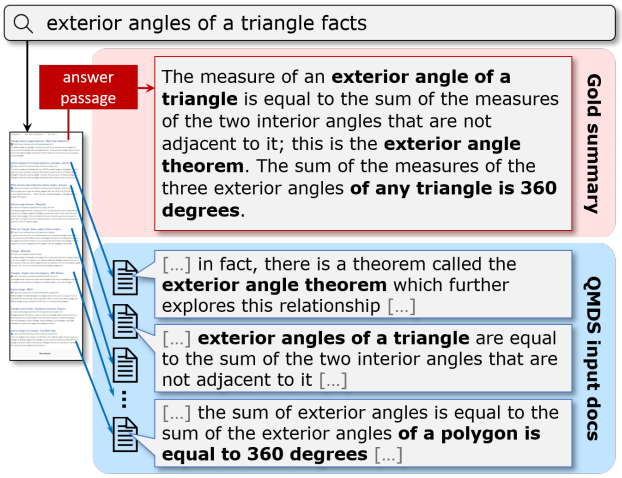


Figure 1: Sample from our QMDSIR dataset, which illustrates how a set of retrieved documents based on a query can provide complementary information that can be used to reconstruct the information in the answer passage (used as gold summary).

Two new data augmentation methods. Recently, multiple new MDS datasets have been introduced: A large-scale dataset by Liu et al. (2018) named WikiSum, and a smaller MDS by Fabbri et al. (2019). Even though WikiSum also includes the topic of the article as query, it is mostly used to train MDS models (Liu et al. 2018; Liu and Lapata 2019), since a topic is more generic to be used as an information seeking query. To this end, we introduce two new data augmentation methods to enable large-scale training of QMDS models. In the first method, we restructure the single-document CNN/Daily Mail (CNN/DM) dataset (Hermann et al. 2015) to create a new QMDS dataset by chunking the documents into small documents of paragraphs and use the title of the article as a query. We refer to this dataset

as QMDSCNN, which has ∼300K samples. For the second method, we mine real-user web queries, top ranked web documents, and answer passages from the search log of Bing (see Fig. 1). We consider the answer passage returned by Bing, which is extracted from one of the top ranked documents as the summary and the rest of the documents as input documents forming our second QMDS dataset. We call this dataset as QMDSIR, which has ∼100K samples. These two new datasets have complementary properties: QMDSCNN has manually written summaries and noisy queries, while the QMDSIR has real queries but automatically generated summaries. Thus, we combine these two datasets to obtain a balanced set of high-quality augmented data, which we used to train our novel, large-scale QMDS models.

Novel models for query-focused MDS task.[1] Liu and Lapata (2019) presented a hierarchical encoder-decoder transformer MDS model with attention layers and incorporated the query by simply concatenating to the top-ranked document. Focusing on building a better abstractive end-to-end neural network-based QMDS model, we introduce **HEROSumm**: **H**i**E**rarchical Que**R**y focused **O**rder-aware multi-document **Summ**arization model, extending the model in Liu and Lapata (2019) with three novel components: (a) ***Hierarchical Encoding***: unlike previous work, which uses a single global representation of the multi-document encoder during the decoding of the summary, we use both the local and global representations from the encoder during decoding; (b) ***Ordering Component***: The QMDS model of (Liu and Lapata 2019) receives the rank order of documents as input from an external module. If the order information is incorrect, it can adversely affect the QMDS model's performance. Hence, to eliminate this cascading error effect, we introduce a new document ordering module that learns the ordering pattern while training the QMDS model parameters end-to-end. (c) ***Query Component***: Unlike previous work, which prepends the query to top document during encoding, we enrich our QMDS model with an additional transformer component that encodes the query. The decoder then attends the local and/or global layers of the multiple document encoders which are conditioned on the query output encoding.

Our quantitative evaluations show that the HEROSumm model, which includes new QMDS focused components, can generate more accurate summaries than the baseline. We also demonstrate that neural models trained on the QMDSCNN and QMDSIR datasets constructed with our data augmentation methods show promising attributes of transferability compared to the models trained on the WikiSum dataset, when tested on real QMDS datasets with summaries written by humans (DUC 2006 and 2007). We further validate the superiority of our data augmentation methods via human evaluation studies along multiple attributes.

2 Related Work

Earlier MDS Research. Earlier extractive MDS work have used various approaches including maximum marginal relevance (MMR) to reduce redundancy (Carbonell and Goldstein 1998), clustering based on topic detection (Radev et al. 2004), graph-based (Erkan and Radev 2004) or hierarchical LDA-style models (Haghighi and Vanderwende 2009), and variants of query-focused summarization (Dang 2005), that orient the summary around a given query (Daumé III and Marcu 2006; Zhao, Wu, and Huang 2009). Earlier abstractive MDS focused on template- and planner-based (McKeown and Radev 1995; Radev and McKeown 1998; Barzilay, McKeown, and Elhadad 1999) and graph-based methods (Ganesan, Zhai, and Han 2010).

Recent MDS Research. Recent neural SDS models have shown significant improvements on both extractive (Nallapati, Zhou, and Ma 2016; Cheng and Lapata 2016; Narayan, Cohen, and Lapata 2018) and abstractive (Rush, Chopra, and Weston 2015; Chopra, Auli, and Rush 2016; Nallapati et al. 2016; Celikyilmaz et al. 2018; Chen and Bansal 2018; Gehrmann, Deng, and Rush 2018) setups. However, MDS models with neural networks are limited by the unavailability of large-scale MDS datasets. Zhang, Tan, and Wan (2018) adapts a state-of-the-art SDS model for MDS. Feigenblat et al. (2017) introduces a extractive-based QMDS model using the Cross-Entropy method. Baumel, Eyal, and Elhadad (2018) introduces query relevance to adapt SDS model to QMDS. Lebanoff, Song, and Liu (2018) exploits the MMR method to fuse disparate sentences from multi-document inputs. Liu and Lapata (2019) introduced a hierarchical transformer model to better encode global and local aspects in multiple documents. In this work, focusing on the *coherency* aspect of summaries, we design components to attend the query and the local and global aspects of documents better, while tracking the *ordering* information to generate more accurate and focused summaries.

MDS Datasets. Recent introduction of large-scale MDS datasets, WikiSum (Liu et al. 2018), Multi-News (Fabbri et al. 2019), Wikipedia Current Events (Ghalandari et al. 2020), set a promising direction for developing powerful neural network models. Focusing on *query* focused summarization of more realistic scenarios, we constructed two new large-scale QMDS datasets, based on popular single document CNN/DM dataset and real search query logs.

3 Two New QMDS Datasets

3.1 QMDSCNN Dataset

CNN/Daily Mail (CNN/DM) is a commonly used SDS dataset (Hermann et al. 2015). The documents are online news articles and the summaries are human written highlights of corresponding articles. We use the scripts provided by See, Liu, and Manning (2017) to obtain this dataset. We present here step-by-step instructions for converting the CNN/DM SDS dataset into a QMDS dataset:

Step-1: Generate a query per document. We use the title of news article as the query to enable a query-focused setup.

Step-2: Chunk documents. Each news article has multiple small paragraphs (approximately 20 small paragraphs), and the sentences in the summary span across these small paragraphs. We randomly group these small paragraphs into chunks (one to four paragraphs per chunk), each chunk forming a new document. In essence, we split the original

[1] Code: https://github.com/ramakanth-pasunuru/QmdsCnnIr

Statistics	Train	Val	Test
QMDSCNN (# samples)	287,113	13,368	11,490
- Avg. # documents	6.5	6.5	6.5
- Avg. Doc. length (# tokens)	355	346	353
- Avg. Query length (# tokens)	13.8	14.5	14.2
QMDSIR (# samples)	82,076	10,259	10,260
- Avg. # documents	5.8	5.4	5.5
- Avg. Doc. length (# tokens)	1,291	1,402	1,379
- Avg. Query length (# tokens)	6.2	6.2	6.2

Table 1: QMDSCNN and QMDSIR statistics.

article into anywhere from one to four smaller documents, composing new **query-documents-summary triplets**.

Step-3: Create new documents from documents on the same topic. CNN/DM dataset contains several documents on similar or the same topic, mostly written by different newsgroups on the same day. Our goal is to collate these documents of similar topics and select chunks from them to append to our new triplets datasets as follows:[2] We take the entire CNN/DM dataset and index all the chunks with BM25 (Robertson and Walker 1994).[3] For each newly constructed query-documents-summary triplet, we take the title of the summary as query, send to the BM25 search engine, which returns chunks from the entire dataset related to the title (as query). We take the top four chunks and append to the original query-documents-summary triplet as new documents. We provide details on the data curation pipeline with example triplets in the arXiv version of this paper.

Table 1 presents the statistics of QMDSCNN dataset. The average number of documents and document length are roughly same across train/val/test sets. Each triplet sample contains around 5-8 documents, from which four documents are retrieved using BM25 as described previously.

Is QMDSCNN dataset suitable for QMDS? Firstly, an accurate abstractive summary should be entailed by the input document and contain only the salient information (Guo, Pasunuru, and Bansal 2018). Specifically for the QMDS task, the query should also be entailed by the summary. Since the documents along with their titles and summaries are all written by humans in the CNN/DM dataset, we assume that the summaries should reflect the title, as well as each summary should be entailed by its corresponding document. We extend the document list of a given query-documents-summary triplet with additional chunks as new relevant documents. Since these relevant documents are retrieved based on the relatedness to the query (title of the summary), they extend the entailment chain such that the abstractive summary of a triplet is also entailed by the corresponding augmented documents.

Secondly, a good summary should contain sentences that span across multiple documents. We measure this by taking a summary and corresponding documents from a query-documents-summary triplet (excluding the retrieved documents), and align each sentence in the summary to one of the documents. We found that there are many triplets whose summary spans multiple documents, thus, enabling multi-document properties. Statistics and additional analysis are in the arXiv version of this paper.

3.2 QMDSIR Dataset

The QMDSIR contains queries that are issued by actual search engine users. This is more realistic than using the titles of articles as queries as in the WikiSum dataset. We follow these steps to construct the QMDSIR dataset:

Step-1: Sample search logs. We randomly sample English queries from Bing search logs in the United States, during the first six months of 2019. Only queries that have natural language answers returned and the answer passages that received positive user feedback are kept.

Step-2: Capture summary text and documents. For each posed-query, we collect the top 10 ranked documents from Bing and the displayed answer passage. The answer passage is extracted from one of the top ranked documents by Bing's production QA system, which is a constantly updated state-of-the-art neural-based single-document extractive summarization model. We use this answer passage as the target summary. We identify the document from which the answer passage is extracted, and omit that document from the candidate documents to enforce the needs of MDS.

Step-4: Construct dataset. The query, the extracted answer passage as summary, and the rest of the top-ranked documents represent the triplets of our QMDSIR dataset (see Table 1 for statistics).

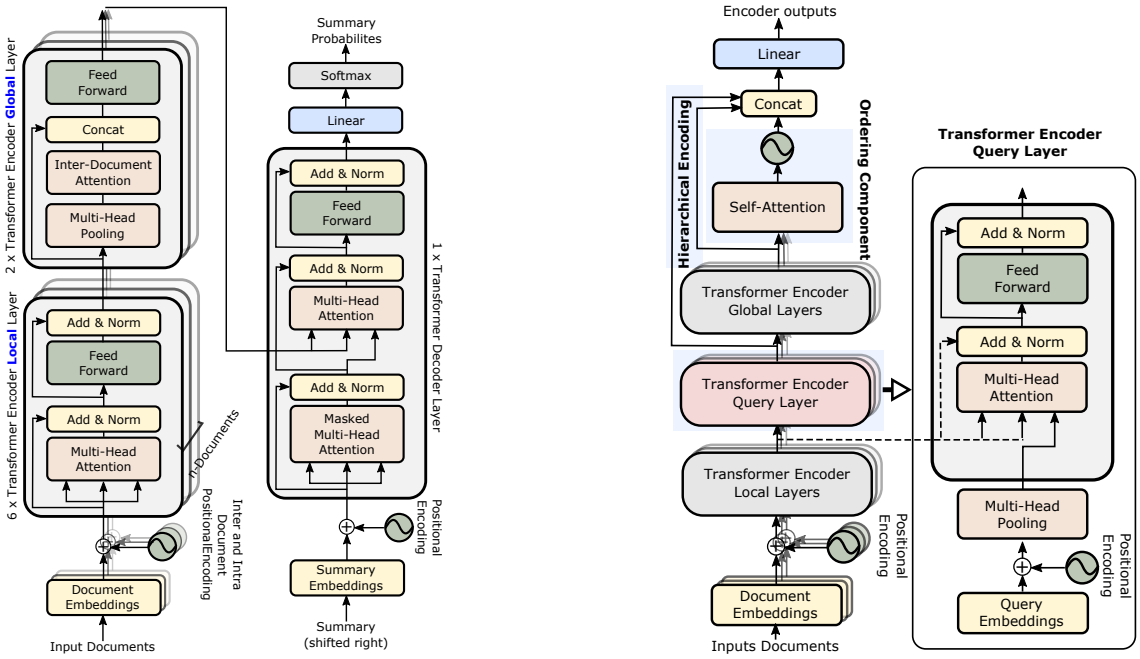
Is QMDSIR dataset suitable for QMDS? Since we use real search query logs, the documents in a triplet are closely related to the query with a potential to answer the query, however, they may or may not contain the direct answer. As shown in Figure 1, collectively the documents may include content to form a summary that can answer the query. This makes our dataset more abstractive in nature as a QMDS model will need to recover and generate the answer passage (summary) using the query and all the other (top-ranked) documents in the triplet.

4 Models

Notation. Our QMDS datasets comprise of instances of triplets of query-documents-summary, $[q, \{D_i\}_i^N, y]$, representing the query q, list-of-documents $\{D_i\}_i^N$ and the summary text y. Each input document D_i of the triplet, is represented as sequence of tokens, D_i=$\{w_{ij}\}_{j=1}^T$, in which w_{ij} is the j^{th} token in the i^{th} ranked document D_i. We represent the latent representations as follows: let the input to the encoder of the transformer be h_{ij}^0. Then the input and output representations of any transformer encoder block in the l^{th} layer is represented with h_{ij}^{l-1} and h_{ij}^l, respectively.

[2] In the scenario where there are no similar topics, the retrieved documents are still useful to simulate the use case of QMDS for presenting the search results, where the returned document set contains both relevant and irrelevant documents.

[3] BM25 is a ranking function used by search engines to estimate the relevance of documents to a given search query.



(a) Baseline encoder-decoder QMDS model.

(b) The HEROSumm encoder.

Figure 2: Comparison of (a) baseline and (b) HEROSumm model with three new components that extends the baseline QMDS model: Hierarchical Encodings, Ordering Component and Query Encoder, enlarged on the right of (b). Unlike baseline model, the HEROSumm decoder attends to both the local and global layers.

4.1 Baseline QMDS Model

Our baseline is similar to the previous work of Liu and Lapata (2019),[4] in which multiple documents are first separately encoded via transformer local encoder layers. Next, we add global transformer layers with multi-head pooling and inter-paragraph attention to allow each document to be aware of the information present in the other documents. Later, we use the output of the global encoder as the encoder context for the transformer decoder layers (see Fig. 2a). In this baseline, we append the query to the first document. Also, we encode the document ranking information in the form of positional encoding which is obtained from a separate document ranker model (Liu and Lapata 2019).

Document Encoding. Each word token w_{ij} in each document D_i is mapped to an embedding vector w^e_{ij}. Since transformers have no sequence information, we use position encoding embeddings similar to Vaswani et al. (2017). Different from SDS models, for MDS we encode both the *inter* and *intra* document positions of each word token w_{ij}. For this, we use two positional encoders, one for inter-document level, representing the order of the document and another for intra-document level, representing the position of the token in the document. Then the positional encoding of a token w_{ij} is concatenation of the inter p^e_i and intra p^e_j document position encodings, respectively. Finally, the input to the transformer h^0_{ij} is represented as: $h^0_{ij} = w^e_{ij} + [p^e_i; p^e_j]$, where $[;]$ presents the concatenation operation.

[4]Liu and Lapata (2019) considered their model in a MDS setup, however, we view it as a simple QMDS model.

Local Transformer Layer. We use the same transformer layer proposed in Vaswani et al. (2017) as our local transformer layer. This layer has the traditional multi-head attention module, feed-forward network, and layer normalization.

Global Transformer Layer. Our global transformer layer is similar to that of Liu and Lapata (2019), which primarily encodes the inter-document context information. This layer has 3 components: (1) **multi-head pooling** to obtain a fixed length document representations; (2) **inter-document attention** to model the dependencies across multiple documents; and (3) **concatenation of the input** with the context from inter-attention followed by a feed-forward network (see Fig. 2a). More details on global transformer layer can be found in Liu and Lapata (2019).

Decoder Transformer. We use the same decoder transformer layer proposed in Vaswani et al. (2017), as shown on the right side of the Fig. 2a.

4.2 HEROSumm Model

Extending the baseline model, our HEROSumm model introduces three new components: a new *encoder* for the query, varying *hierarchical encoding* inputs for the decoder, as well as unsupervised learning of the *order* of the salient concepts to be presented in the generated summary. All these components are targeted on the encoder, so in Fig. 2b we are only showing the encoder part of the model.

Query Encoder. Unlike the baseline model in which the query text is simply appended to the top-ranked document before sending it to the encoder, we encode the query via

a separate transformer encoder layer. This layer is inserted between local and global layers of the encoder, as shown in Fig. 2b (with an enlarged view provided on the right of the figure). A separate query layer creates a hierarchy of information encoding, i.e., the local layers enable a rich intra-document feature representation, the query layer conditions this local layer features w.r.t. the given query, and the global layer enable the inter-document feature representation on the query conditioned local layers.

Let q_k be the k^{th} token in the query, and h^l_{ij} be the output of j^{th} token in i^{th} ranked document of the last local layer before the query layer. The query input representation (h^q_k) for the query layer is a combination of its token embeddings (q^e_k) and the positional encoding p^q_k, which is defined as $h^q_k = w^q_k + p^q_k$. We encode the query input along with the last local layer output (h^l_{ij}) in the following steps to form our transformer encoder query layer:

$$\begin{aligned} o^1_i &= \text{LN}(h^l_i + \text{MHA}(h^l_i, h^l_i, \text{MHP}(h^q))) \\ o^2_i &= \text{LN}(o^1_i + \text{FFN}(o^1_i)) \end{aligned} \tag{1}$$

where, MHA is multi-head attention, MHP is multi-head pooling (Liu and Lapata 2019) which is applied on full query tokens (h^q), LN is layer normalization, and FFN is feed-forward networks. o^2_i is the output from this layer which is used as input to the transformer encoder global layer.

Hierarchical Encodings. Unlike the baseline model (Liu and Lapata 2019), in which the decoder only attends to the global layer features, the HEROSumm decoder attends to both the output of the local and global layers taking into account both context. Our intuition is that the local layers carry information specific to the individual documents, while the global layers carry information w.r.t. all the documents. Specifically, the decoder utilizes the global properties from all documents by attending over to the output of the global layer. It can also attend to the local layers to focus on the specific aspects of the documents, in which salient information related to the query may be more pronounced. We concatenate the output of the local and global layers and project it through a linear layer, as shown in Fig. 2b top-left.

Self Ordering Transformer Encodings. In QMDS, the rank-order of the list of documents is an important information as it helps the model to weigh in on the documents relevant to the query. Otherwise, focusing equally on all documents makes it very hard for the model to weed out the salient information and also present them in the correct order in the summary. Previous work (Liu and Lapata 2019) introduced a two-stage pipeline to inject the ordering into their model. In the stage-1, a document ranker is trained separately to learn the importance of a document w.r.t. a given query. In the stage-2, they use these importance scores to rank the documents and encode them in the model. However, the errors introduced by the document ranker can potentially have cascading effects on the performance of the final summarization model.

To address this issue, we propose a single-stage model that jointly learns to rank the documents while learning to generate salient summary via our ordering component (see Fig. 2b). Instead of using the positional encoding of the document positions predicted by the document ranker, we ignore the document position embeddings in the initial layer of the transformer, and encode the positional embeddings of the documents at the final layer of the transformer encoder. For this, we use a self-attention module (Lin et al. 2017) to know the importance of each document in the multi-document setup. We then encode this importance information in a novel way via a positional encoding module:

$$\begin{aligned} \text{PE}_{(D_i,2j)} &= \sin(r_i/10000^{2j/d_{model}}) \\ \text{PE}_{(D_i,2j+1)} &= \cos(r_i/10000^{2j/d_{model}}) \end{aligned} \tag{2}$$

where, $\text{PE}(D_i, 2j)$ represents the $2j^{th}$ dimensional positional encoding representation of document D_i, r_i is the importance score assigned for document D_i using the self-attention module, and d_{model} is the model's hidden size. This positional encoding module allows us to convert an unordered importance score into an ordering representation, since unordered score projected on a sinusoidal wave are positioned in an orderly fashion. Finally, we concatenate the final global layer representations of the encoder with the document ordering-based positional encoding representations to form the final encoder representations (Fig. 2b, top-left).

5 Experimental Setup[5]

Datasets. We use three large datasets for training QMDS models: our two datasets QMDSCNN and QMDSIR, described in Sec. 3.1, and the WikiSum. We also use DUC 2006 and DUC 2007 datasets for evaluating our models.[6]

Model Ablations and Training. We experiment with four different ablations of HEROSumm (**HS** in short) model extending the baseline QMDS model of Liu and Lapata (2019) with only hierarchical encodings, only the ordering component, and with the query encoding. **HS-Joint**, our full model, combines two or three of these components depending on the type of the dataset used in the experiments.

Evaluation Metrics. We use ROUGE (Lin 2004), i.e., ROUGE-1, ROUGE-2, and ROUGE-L as our automatic evaluation metrics. We report sentence-level ROUGE F1 scores for all large-scale datasets. For DUC datasets, we report sentence-level ROUGE recall with 250 word limit.

6 Results

We present empirical results of our proposed models on various datasets. We first report on three large-scale QMDS augmented datasets: WikiSum, QMDSCNN, and QMDSIR, to understand how well various models fit the augmented data. Validating the superiority of our proposed models and augmentation methods, we also show transfer results of training on our augmented datasets by using DUC 2006 and 2007 (two human-annotated real QMDS datasets) as test sets.

6.1 Results on Accuracy

WikiSum Dataset. Table 2 shows the results on the WikiSum dataset. We observe that both hierarchical encodings

[5]Due to space constraints and no supplementary allowed in AAAI rules, we provide more details in the arXiv version.

[6]https://www-nlpir.nist.gov/projects/duc/data.html

Model	R-1	R-2	R-L
Liu and Lapata (2019)*	38.03	24.68	36.20
HS w/ Hierarchical Encodings	38.14	24.88	36.33
HS w/ Ordering Component	**38.57**	**25.13**	**36.71**
HS w/ Query Encoding	35.70	21.86	33.70
HS-Joint Model	38.37	24.90	36.52

Table 2: Performance of our baseline and variations of HEROSumm (HS) model on WikiSum dataset. R-1, R-2, and R-L denote sentence-level ROUGE-1, ROUGE-2, and ROUGE-L, respectively. $\star$ is the reproduced result from the code provided by Liu and Lapata (2019).

and ordering methods improve the performance of the model in comparison to the corresponding baseline.[7][8] However, the addition of separate query encoding did not improve the results, in fact, they become worse. This can be explained by the fact that this dataset may not be well suited for evaluating the QMDS models since the queries are constructed from the title of the Wikipedia article while the summaries are taken as the first paragraph of the article. Thus, neither the queries nor the summaries are natural nor constructed to reflect the properties of a high-quality QMDS dataset. Finally, we combine the hierarchical and ordering methods to form the joint model (see Fig. 2) which again performs significantly better than the baseline with $p < 0.05$ in all metrics.

QMDSCNN Dataset. Table 3 presents the evaluation results of our baseline and three of our HEROSumm model variations (using hierarchical encodings, ordering, and query encoding) on the new QMDSCNN dataset. We observe that both HS models with hierarchical encodings and query-based methods perform significantly better than the baseline, however, HS with ordering method did not work well on this dataset.[9] For this experiment, our HS-Joint model combines the hierarchical encodings and the query encoder components. We observe that HS-Joint model is significantly better in context match accuracy than the baseline with $p < 0.01$. Our HS with hierarchical encodings method outperformed the HS-Joint model. This can be attributed to the fact that hierarchical modeling of local and global information is more crucial for this dataset while summaries don't share complementary information with the query.

QMDSIR Dataset. Table 4 shows the results on QMDSIR dataset, comparing our model ablations against the baseline. For this experiment, our HS-Joint model is a combination of hierarchical encodings and query encoding components. We observe that HS with query encodings method performs significantly better than the baseline (with $p <$ 0.01 on ROUGE-1 and ROUGE-2 metrics, and $p < 0.05$ on ROUGE-L), suggesting that this dataset enables efficient use of the queries by the QMDS models. Overall, we achieve best results with our HS-Joint model in comparison to our baseline and other HS ablations with $p<0.01$ in all metrics.

[7]HEROSumm (HS) with hierarchical encodings and HS with ordering method are statistically significantly better than baseline with $p < 0.05$ and $p < 0.01$, respectively, in all metrics.

[8]We initially tried the random ranking order of input documents, and it performed worse than original order (baseline in Table 2), which in turn performed lower than our ordering component.

[9]Both hierarchical encodings and query-based methods perform significantly better than baseline with $p<0.01$ in all metrics. Ordering method also performed well on ROUGE-1/L ($p<0.01$).

Model	R-1	R-2	R-L
Liu and Lapata (2019)	36.31	15.40	33.38
HS w/ Hierarchical Encodings	**37.88**	**16.36**	**35.23**
HS w/ Ordering Component	36.95	14.95	34.34
HS w/ Query Encoding	36.96	16.05	34.37
HS-Joint Model	37.09	16.33	34.45

Table 3: Accuracy results on QMDSCNN dataset.

Model	R-1	R-2	R-L
Liu and Lapata (2019)	43.60	21.88	39.40
HS w/ Hierarchical Encodings	43.37	21.64	39.21
HS w/ Ordering Component	39.37	18.79	35.61
HS w/ Query Encoding	44.11	22.62	39.93
HS-Joint Model	**45.53**	**23.44**	**41.15**

Table 4: Accuracy results on QMDSIR dataset.

6.2 Results on Transfer Learning

We use the DUC 2006 and 2007 datasets for transfer learning experiments with two scenarios. In the first scenario, we train on the 3 large QMDS datasets and show transfer results on the DUC 2006 and 2007 datasets. In the second one, we finetune models from the first scenario on DUC 2006, and then test on DUC 2007. We evaluate on quantitative and qualitative metrics using ROUGE and human evaluations, respectively. In both scenarios, we compare results of the baseline model (Liu and Lapata 2019) to our HS-Joint model, which is the last row in Table 2, 3, & 4. Our data augmentation methods are not specific to solve DUC datasets, but rather aim to improve QMDS in general, where DUC is one of the standard evaluation sets on which we show improvements via transfer setup. We believe our data augmentation methods would be useful for the community in creating larger-scale training datasets for QMDS.

Impact of our data augmentation methods. Table 5 shows results when DUC 2006 and DUC 2007 datasets are used as test sets and compare our HS-Joint models against the baseline models. We report recall scores with 250 word length. Based on pre-training experiment results on DUC 2006 in Table 5(a) and on DUC 2007 in Table 5(b), our data augmentation methods perform better than training on the WikiSum dataset by a large margin. Our baseline models trained on the combined datasets, QMDSCNNIR, outperform all other baseline models. However, on the HS-joint models, QMDSCNNIR is not better than individual data augmentation methods. This suggests that we might also need better weighted sampling or curriculum learning when we combine these two datasets, which we leave for future work. However, we believe that the individual contributions of our two

Model	Dataset	(a) **DUC 2006 test set** R-1	R-2	R-L	R-SU4	(b) **DUC 2007 test set** R-1	R-2	R-L	R-SU4	(c) **DUC 2007 test set** R-1	R-2	R-L	R-SU4
Baseline	No-Pretraining	-	-	-	-	-	-	-	-	10.01	1.42	9.80	3.18
	WikiSum	24.00	4.28	22.72	8.16	23.42	4.41	22.17	8.05	34.34	6.35	32.07	11.42
	QMDSIR	29.65	3.83	27.93	9.63	29.35	3.75	27.45	9.55	32.81	4.15	30.54	10.53
	QMDSCNN	30.45	6.13	28.61	10.39	31.72	**7.07**	29.69	11.34	36.80	**7.36**	34.49	12.53
	QMDSCNNIR	**30.57**	**6.17**	**28.88**	**10.57**	**32.33**	6.98	**30.50**	**11.63**	**37.07**	**7.36**	**34.62**	**12.60**
HS-Joint	No-Pretraining	-	-	-	-	-	-	-	-	18.80	2.40	18.17	5.94
	WikiSum	22.96	4.09	21.76	7.89	22.91	4.45	21.74	7.92	29.97	4.22	27.98	9.10
	QMDSIR	30.17	4.01	28.31	9.79	29.74	3.74	27.83	9.71	31.33	4.02	29.29	10.10
	QMDSCNN	**31.14**	**6.28**	**29.28**	10.90	**34.14**	**7.60**	**32.08**	**12.50**	**38.31**	**7.64**	**35.65**	**13.26**
	QMDSCNNIR	30.83	6.18	29.17	**10.91**	33.13	7.37	31.05	12.04	36.26	6.49	33.79	12.34

Table 5: Transfer results where each model is trained on various datasets, and tested respectively on (a) DUC 2006 and (b) DUC 2007. In (c), the pre-trained models of (b) are then fine-tuned on DUC 2006 dataset.

data augmentation methods (QMDSCNN and QMDSIR) are still useful. We observe similar behavior on the DUC 2007 after fine-tuning in Table 5(c).

Impact of new QMDS components. Compared to the baseline model, our HS-Joint models, which incorporate novel QMDS focused components, yield much better results when trained on datasets constructed with data augmentation methods and tested on real human datasets as shown in Table 5. Results support that with better data augmentation and a much better transformer architecture, we can build more accurate models with higher transfer capabilities.[10]

Human evaluation. We also evaluate our data augmentation methods using head-to-head human evaluations on Amazon Mechanical Turk (AMT). We compare summaries generated by two baseline models: one trained on the WikiSum (WIKI) and another one on the QMDSCNNIR (CB) dataset, combining our two new datasets. We generate samples from DUC 2006 and DUC 2007 test dataset, and each sample is evaluated by 3 judges. For DUC 2007, we use the same models fine-tuned on the DUC 2006 dataset as explained earlier. For each test dataset, we ask the turkers to choose between the two model summaries that answer the given query based on 5 different aspects:[11] (1) *Informativeness*: which summary is better in terms of answering the query better? (2) *Non-redundancy*: which summary is better in terms of repeating less of the same ideas? (3) *Coherence*: which summary is better in terms of expressing ideas in the clearest manner fluently? (4) *Order*: which summary is better at presenting the information in the logical order? (5) *Focus*: which summary is better in terms of only sharing the main ideas with no extra superfluous details? We also ask the turkers to compare the summaries on overall quality. We chose the turkers who are located in the USA and UK, have at least $10,000$ approved HITs, and have an approval rate of greater than 98%. We pay \$0.5 for a HIT. The results are as shown in Table 6, where on DUC 2006, our data augmentation CB method yields much better results compared to the one on WikiSum in all aspects.[12] On DUC 2007, a fine-tuning setup, we still see that our method is better in all aspects except focus.[13] Overall, these human evaluations also suggest that our augmentation methods are better than previous work (WikiSum).

Criteria	DUC 2006 WIKI	CB	=	DUC 2007 WIKI	CB	=
Informativeness	30	**107**	12	52	**63**	20
Non-redundancy	27	**110**	12	51	**58**	25
Coherence	27	**112**	11	55	**59**	20
Order	25	**112**	11	56	**63**	12
Focus	21	**117**	12	**61**	59	13
Overall	23	**103**	24	46	**53**	35

Table 6: Human evaluation between baseline model trained on WikiSum (WIKI) and QMDSCNNIR (CB) datasets. '=' denotes no difference between the two.

7 Conclusions

To support research on query-focused multi-document summarization task, we introduce two new data augmentation methods using existing and new data sources. We further introduce a new transformer encoder-decoder model that extends the baseline models with new components to encode the queries together with multiple documents in a hierarchical setting. New components enrich the information provided to the decoder that generates focused summaries. We show that summaries generated by the models trained on augmented datasets are more accurate compared to the existing datasets. Additionally, our best model can generate summaries that are coherent and contain specific information related to the query with better order of events.

[10] The current SOTA extractive QMDS model (Roitman et al. 2020) achieves R-1/R-2/R-SU4 scores of 43.94/10.09/15.96 on DUC 2006 and 46.02/12.53/17.91 on DUC 2007. However, it is not strictly comparable with our end-to-end abstractive QMDS models.

[11] During AMT evaluation, we also show one of the gold summaries without providing the original documents.

[12] Our CB augmentation method is statistically significantly better than WikiSum in all 5 aspects with $p < 0.001$ based on bootstrap test (Noreen 1989; Efron and Tibshirani 1994).

[13] Even though our method performed lower on the focus aspect, the difference is very low (lowest w.r.t. all other aspects).

Acknowledgments

We thank the reviewers for their helpful comments. We thank Tong Wang at Microsoft Turing for helping create QMDSIR. We also thank Paul Bennett, Tobias Schnabel, Woon Sang Cho, Chen Qu, and Jiawei Wu for helpful discussions. This work was partially supported by NSF-CAREER Award 1846185 and a Microsoft PhD Fellowship.

References

Barzilay, R.; McKeown, K.; and Elhadad, M. 1999. Information fusion in the context of multi-document summarization. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*, 550–557.

Baumel, T.; Eyal, M.; and Elhadad, M. 2018. Query focused abstractive summarization: Incorporating query relevance, multi-document coverage, and summary length constraints into seq2seq models. *arXiv preprint arXiv:1801.07704* .

Carbonell, J.; and Goldstein, J. 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*, 335–336.

Celikyilmaz, A.; Bosselut, A.; He, X.; and Choi, Y. 2018. Deep communicating agents for abstractive summarization. In *NAACL*.

Chen, Y.-C.; and Bansal, M. 2018. Fast Abstractive Summarization with Reinforce-Selected Sentence Rewriting. In *Proceedings of ACL*.

Cheng, J.; and Lapata, M. 2016. Neural summarization by extracting sentences and words. In *ACL*.

Chopra, S.; Auli, M.; and Rush, A. M. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *HLT-NAACL*.

Dang, H. T. 2005. Overview of DUC 2005. In *Proceedings of the document understanding conference*, volume 2005, 1–12.

Daumé III, H.; and Marcu, D. 2006. Bayesian query-focused summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, 305–312. Association for Computational Linguistics.

Efron, B.; and Tibshirani, R. J. 1994. *An introduction to the bootstrap*. CRC press.

Erkan, G.; and Radev, D. R. 2004. LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research* 22: 457–479.

Fabbri, A. R.; Li, I.; She, T.; Li, S.; and Radev, D. R. 2019. Multi-News: a large-scale multi-document summarization dataset and abstractive hierarchical model. In *ACL*.

Feigenblat, G.; Roitman, H.; Boni, O.; and Konopnicki, D. 2017. Unsupervised query-focused multi-document summarization using the cross entropy method. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 961–964.

Ganesan, K.; Zhai, C.; and Han, J. 2010. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In *COLING*.

Gehrmann, S.; Deng, Y.; and Rush, A. M. 2018. Bottom-up abstractive summarization. In *EMNLP*.

Ghalandari, D. G.; Hokamp, C.; Pham, N. T.; Glover, J.; and Ifrim, G. 2020. A Large-Scale Multi-Document Summarization Dataset from the Wikipedia Current Events Portal. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*.

Grusky, M.; Naaman, M.; and Artzi, Y. 2018. Newsroom: A Dataset of 1.3 Million Summaries with Diverse Extractive Strategies. In *NAACL*.

Guo, H.; Pasunuru, R.; and Bansal, M. 2018. Soft Layer-Specific Multi-Task Summarization with Entailment and Question Generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 687–697. Melbourne, Australia: Association for Computational Linguistics.

Haghighi, A.; and Vanderwende, L. 2009. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 362–370.

Hermann, K. M.; Kocisky, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; and Blunsom, P. 2015. Teaching machines to read and comprehend. In *NeurIPS*, 1693–1701.

Lebanoff, L.; Song, K.; and Liu, F. 2018. Adapting the Neural Encoder-Decoder Framework from Single to Multi-Document Summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 4131–4141.

Lin, C.-Y. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text summarization branches out*, 74–81.

Lin, Z.; Feng, M.; Santos, C. N. d.; Yu, M.; Xiang, B.; Zhou, B.; and Bengio, Y. 2017. A structured self-attentive sentence embedding. In *ICLR*.

Liu, P. J.; Saleh, M.; Pot, E.; Goodrich, B.; Sepassi, R.; Kaiser, L.; and Shazeer, N. 2018. Generating wikipedia by summarizing long sequences. In *ICLR*.

Liu, Y.; and Lapata, M. 2019. Hierarchical Transformers for Multi-Document Summarization. In *Proc. of ACL*.

McKeown, K.; and Radev, D. R. 1995. Generating summaries of multiple news articles. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, 74–82.

Nallapati, R.; Zhou, B.; dos santos, C. N.; Gulcehre, C.; and Xiang, B. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *CoNLL*.

Nallapati, R.; Zhou, B.; and Ma, M. 2016. Classify or select: Neural architectures for extractive document summarization. *arXiv preprint arXiv:1611.04244* .

Napoles, C.; Gormley, M.; and Durme, B. V. 2012. Annotated Gigaword. *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction* .

Narayan, S.; Cohen, S. B.; and Lapata, M. 2018. Ranking Sentences for Extractive Summarization with Reinforcement Learning. In *NAACL*, 1747–1759.

Noreen, E. W. 1989. *Computer-intensive methods for testing hypotheses*. Wiley New York.

Qazvinian, V.; and Radev, D. R. 2008. Scientific Paper Summarization Using Citation Summary Networks. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, 689–696. Manchester, UK: Coling 2008 Organizing Committee.

Radev, D. R.; Jing, H.; Styś, M.; and Tam, D. 2004. Centroid-based summarization of multiple documents. *Information Processing & Management* 40(6): 919–938.

Radev, D. R.; and McKeown, K. R. 1998. Generating natural language summaries from multiple on-line sources. *Computational Linguistics* 24(3): 470–500.

Robertson, S. E.; and Walker, S. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR'94*, 232–241. Springer.

Roitman, H.; Feigenblat, G.; Cohen, D.; Boni, O.; and Konopnicki, D. 2020. Unsupervised Dual-Cascade Learning with Pseudo-Feedback Distillation for Query-Focused Extractive Summarization. In *Proceedings of The Web Conference 2020*, 2577–2584.

Rush, A. M.; Chopra, S.; and Weston, J. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*.

See, A.; Liu, P. J.; and Manning, C. D. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *ACL*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *NeurIPS*, 5998–6008.

Zhang, J.; Tan, J.; and Wan, X. 2018. Adapting neural single-document summarization model for abstractive multi-document summarization: A pilot study. In *Proceedings of the 11th International Conference on Natural Language Generation*, 381–390.

Zhao, L.; Wu, L.; and Huang, X. 2009. Using query expansion in graph-based approach for query-focused multi-document summarization. *Information processing & management* 45(1): 35–41.