Generating a Question as the Theme Heading for Multiple Recommended Articles in E-bibliotherapy

YUNXING XIN, Tsinghua University, China
LING FENG, Tsinghua University, China

Bibliotherapy has proven its effectiveness in dealing with mental disorders through the long-term clinic practice and academic research. With respect to the increasing adolescent phycological problems, we built a reading recommendation system *TeenRead* to assist carrying out online e-bibliotherapy for teens' stress alleviation. While reducing manual readings' selection effort through automatic recommendation, we further propose to generate a question as the theme heading for multiple recommended articles. Such a heading shall not only convey the essential message of the recommended articles, but also attract and guide teens to read the articles. More importantly, a heading directly responding to teens' questions could enable to generate emotional resonance and thus willingness to pursue the reading. In this article four solutions based on the neural encoder-decoder model are presented to tackle the problem. For model training and testing, we construct a large-scale topic-specific naturally annotated QAs dataset *TeenQA*. Due to the flexibility of question expressions, we incorporate three types of performance metrics to examine the lexical similarity, human consensus, and semantic similarity of the generated question-like theme headings. Our experimental results show that our ED-SoCF solution performs best, and its performance on semantic similarity metrics is comparable to that of humans.

CCS Concepts: • **Information systems** → **Question answering**; *Recommender systems*; • **Computing methodologies** → **Natural language generation**; *Neural networks*; • **Applied computing** → *Health care information systems*; Psychology;

Additional Key Words and Phrases: E-bibliotherapy, theme heading generation, question-like, encoder-decoder, dataset

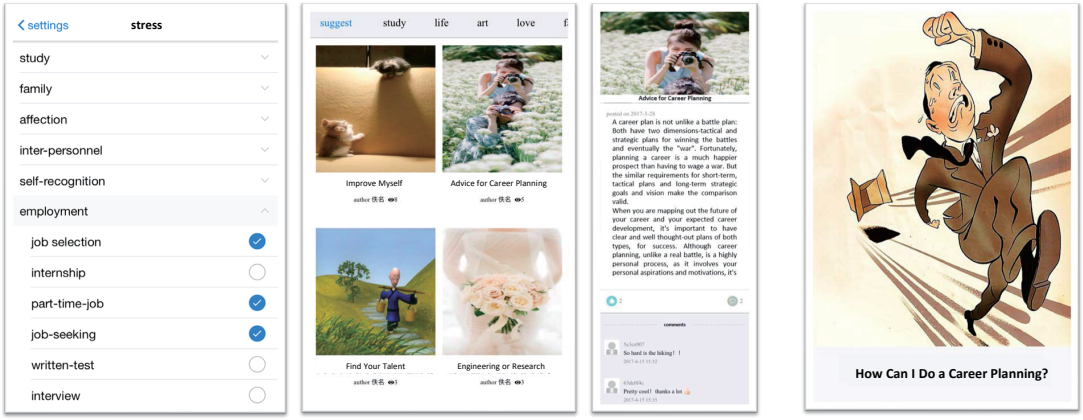
ACM Reference Format:
Yunxing Xin and Ling Feng. 2018. Generating a Question as the Theme Heading for Multiple Recommended Articles in E-bibliotherapy. *ACM Transactions on Information Systems* , (January 2018), 34 pages. https://doi.org/0000001.0000001

Authors' addresses: Yunxing Xin, Tsinghua University, Department of Computer Science and Technology, Centre for Computational Mental Healthcare Research, Institute of Data Science , Beijing, 100084, China, xinyx16@mails.tsinghua.edu.cn; Ling Feng, Tsinghua University, Department of Computer Science and Technology, Centre for Computational Mental Healthcare Research, Institute of Data Science, Beijing, China, fengling@mail.tsinghua.edu.cn.

1 INTRODUCTION

1.1 Background

With the rapid development of economy and society, teens are facing various psychological stress coming from study, family, love, peer relation, self-cognition, and so on. Bibliotherapy has been proved to be an effective treatment method to cope with psychological stress [Burns 1980; Crothers 1917; McKenna et al. 2010]. It involves the reading of specific literature with the purpose of prevention, healing and rehabilitation. Through resonance, intimation, and/or apperception at the highest, readers could develop strength and courage to cope with their stress or mental problems [Burns 1980]. As an adjunct therapy to psychological treatment, bibliotherapy is effective to people of all ages. But this traditional bibliotherapy based on paper materials is ill-fitted in the present information era, especially for teens. On the one hand, bibliotherapy requires a lot of professional cares, which is labor demanding and difficult to be carried out by teens themselves. On the other hand, teens, who are deeply pressurized, tend not to actively take the treatment for the self-esteem reason. To address the limitations, the notion of e-bibliotherapy, which combines bibliotherapy with computers and internet, thus arose [Elizabeth et al. 2016]. [Phornphatcharaphong 2012] developed an e-bibliotherapy system that helps youth to share books and build friendship networks. [Xin et al. 2017] built an adolescent reading recommendation and sharing mobile application *TeenRead*, which weekly recommends 4 articles to users based on their stress categories and subcategories, as shown in Fig. 1(a). *TeenRead*'s reading materials are from authorized or active users, who wish to share literature as a means of personal growth and development.



(a) Users first set their stress, then based on that *TeenRead* recommends 4 articles.

(b) The generated theme heading in question form.

Fig. 1. The interfaces of *TeenRead*.

TeenRead's automatic recommendation of appropriate articles to stressful teens alleviates manual readings selection efforts. This made one step towards online e-bibliotherapy. Beyond that, for the sake of good reading experiences and effective bibliotherapy, a proper theme heading about the recommended multiple

articles is also important and desirable. Such a heading shall not only convey the essential message of the recommended articles, saving teens efforts in identifying the useful information, but also attract and guide teens to read the articles.

Psychological studies show that devising questions before reading can help attract attentions and improve active comprehension of complex short stories [Singer and Donlan 1982]. More importantly, considering the mission of *TeneRead* is to assist teens to ease their stress by reading, the most effective way, which we believe can achieve that, is to react to and address teens' problems. If the theme reading is presented in a question (like *How to get along with parents who do not understand me?*, *what to do after quarreling with friends?*, *How to pass an exam?*), showing care and concern about stressful teens, they may feel resonance in emotion, and are thus willing to pursue the reading.

To this end, this study explores how to generate a theme heading in a question form for the recommended multiple articles (Fig. 1(b)).

1.2 Challenges

In the literature, headline generation and question generation is treated as two separate tasks with different application domains [Banko et al. 2000; Dorr et al. 2003; Du et al. 2017; Duan et al. 2016; Filippova 2010; Gattani 2007; Heilman 2011a; Heilman and Smith 2010; Knight and Marcu 2000b; Sun et al. 2015; Tan et al. 2017; Tang et al. 2017; Unno et al. 2006]. However, generating a question-like theme heading is not explored to our knowledge. For multiple articles, generating a theme heading in question form is a non-trivial task, facing a number of challenges.

- *Task novelty and complexity.* Although question generation (QG) task has been studied for years in the area of education for reading comprehension [Du et al. 2017; Heilman 2011a; Heilman and Smith 2010] and question answering systems in natural language processing and AI fields [Duan et al. 2016; Tang et al. 2017], most of them aim to generate questions from structured data (e.g., knowledge bases [Serban et al. 2016] and concept map [Olney and Person 2012]), or from a single sentence/passage [Chali and Hasan 2015a; Du et al. 2017; Duan et al. 2016; Labutov et al. 2015; Yao et al. 2012; Yuan et al. 2017a; Zhou et al. 2018], whose answer is usually brief and objective. In this study, on the contrary, the answer is subjective, redundant, and spreading over multiple articles, which requires more efforts to find the key information and join them to generate a general question. This paper is the first trial to generate a question from multiple long texts, where the question targets at the issue(s) addressed by the subjective multiple texts.
- *Question type's diversity.* Most of the previous work on QG focused on a specific type of questions, especially fact-based factoid questions [what, when, which, who, whom, where] [Serban et al. 2016], where a strong connection between answers and questions exists. The answers to factoid questions are usually short, containing a short span of words, entity, or sentence. In our task, apart from factoid questions, other types of questions (e.g., causal and explanatory questions [how, why] asking for causal explanations, and methodological questions [what, how should] seeking for problem solutions and advice, etc.), need to be considered. Answers to these questions could be either objective or subjective, and tend to be long.

- *Information coverage.* A theme heading contains only a few words (usually no more than 16 words and punctuation marks), but is requested to cover the important information of multiple articles, each of which can be pretty long. This requires machine understanding, effective information extraction and abstraction techniques, all of which remain to be very tough in the current research.
- *Statement fluency and diversity.* The generated theme heading should be readable with correct morphology and syntax. Besides, for the sake of attractiveness, the theme expression should be various and flexible. We can't imagine how disgusted users would be if all theme headings are in a fixed"*How can I ...*?" style. From this perspective, heading generation based on heuristic rules and templates [Chali and Golestanirad 2016; Heilman 2011b; Labutov et al. 2015; Lindberg et al. 2013] is insufficient for our task.

1.3 Our Work

Inspired by the recent success of neural encoder-decoder models in handling sequence-to-sequence (seq2seq) tasks like machine translation [Bahdanau et al. 2014; Luong et al. 2015a; Sennrich et al. 2016; Sutskever et al. 2014; Wu et al. 2016], text summarization [Chopra et al. 2016; Gu et al. 2016; Lopyrev 2015; Nallapati et al. 2016; Rush et al. 2015], speech recognition [Bahdanau et al. 2016], and video captioning [Venugopalan et al. 2015], we turn to encoder-decoder models for our question-like theme heading generation task. The basic idea is that we first encode the articles into a context vector representation, then based on that the decoder output the theme word by word.

However, construction of such an effective encoder-decoder model relies heavily on a large-scale high-quality dataset. Available QA datasets [Bordes et al. 2015; Joshi et al. 2017; Nguyen et al. 2016; Rajpurkar et al. 2016; Trischler et al. 2016; Yang et al. 2015] are not suitable in our task. On the one hand, the content of those datasets is not specific for teens. On the other hand, their questions are derived from context(s) or facts, and most of the answers are briefly objective. Whereas in our task, we need a novel QA dataset, of which the content and the questions are adapted to teens' stress categories, the questions can be derived from answers without context, and the answers are complex enough to help ease teens' stress. Hence, before the construction of an encoder-decoder model, we need to make efforts to build a large scale suitable dataset.

Overall, the study makes the following three contributions.

- We propose a novel task of question-like theme heading generation from multiple recommended articles for e-bibliotherapy, whose aim is to assist stressful teens to release stress by reading. Four solutions based on the neural encoder-decoder model are presented to tackle the problem: encoder-decoder with summary on outputs (ED-SoO), encoder-decoder with summary on inputs (ED-SoI) and encoder-decoder with summary on contexts with(out) feature-rich embeddings (ED-SoC and ED-SoCF).
- We collect and construct a large-scaled naturally annotated **QA**s dataset TeenQA for model training and testing. Different from the existing datasets, TeenQA is topic-specific, containing **Teen**s' commonly-encountered problems and community-given solutions. In terms of data size, question diversity, answer complexity, and content subjectivity, TeenQA presents more challenges for question-like theme generation task.

- We conduct extensively experiments on three groups of performance metrics to evaluate the lexical similarity, human consensus, and semantic similarity of the generated question-like theme headings from the four solutions. Our experimental results show that ED-SoCF performs best among the four solutions: 27%, 18%, 13% more than ED-SoO, ED-SoI, and ED-SoC on lexical metrics, 48%, 53%, 38% more on consensus metric, and its performance on semantic metrics is comparable to that of humans.

The remainder of the paper is organized as follows. We review related work in the areas of question generation and headline generation in Section 2. The acquisition and analysis of our constructed dataset *TeenQA* are detailed in Section 3. Approaches of generating a question-like theme heading for multiple articles, as well as their performance, are given in Section 4 and 5, respectively. Finally, we conclude the paper and point out future work in Section 6.

2 RELATED WORK

Our work is closely related to existing studies on headline generation and question generation.

2.1 Headline Generation

Headline generation is a task of producing a condensed text summarization over one or multiple documents such as news stories. Headline generation methods can be generally divided into three categories: *extractive*, *abstractive*, and *neural*.

2.1.1 Extractive Headline Generation Methods. In extractive methods, candidate sentences from original documents are extracted and put together to form the headline through sentence compression techniques. Four main lines of extractive headline generation techniques are developed and broadly employed.

(1) *Linguistic Rule-based Methods*

These methods make use of handcrafted linguistic rules for detecting and compressing important parts of documents. The representative example of this method is Hedge Trimmer [Dorr et al. 2003], which built a parse and trim schema, and generated the headline for a news story by removing constituents from the parse tree of the lead (first) sentence of the news article until a certain length threshold is reached. In [Dorr et al. 2003], linguistically motivated techniques guide the choice of what constituents should be removed and retained.

The rule-based methods are simple and lightweight, and do not require prior training on a large corpus. However, as only limited candidate sentences (e.g., the first sentence [Dorr et al. 2003]) are considered, rule-based approaches fail in capturing and exploring complex relations throughout the text for headline generation [Gattani 2007].

(2) *Statistics-based Methods*

The methods exploit statistic models for learning correlations between words in headlines and in documents, and work in a supervised learning setting with a large training corpus. A notable work is [Banko et al. 2000], which used the Naïve Bayes approach to learn the conditional probability of a word appearing in a headline given it appears in the document: $P(w \in H | w \in D) = \frac{P(w \in H \wedge w \in D)}{P(w \in D)}$. To enforce the sentence structure and score candidate headlines, [Banko et al. 2000] further computed the probability of word sequence through a bi-gram language model. The overall probability of a candidate headline H consisting

of word sequence $(w_1, w_2, \cdots, w_n)$ is computed as the product of the likelihood of (1) the terms selected for the headline (2) the length of the resulting headline, and (3) the most likely sequencing of the terms in the content set [Banko et al. 2000]. $P(w_1, w_2, \cdots, w_n|D) = \prod_{i=1}^{n} P(w_i \in H|w_i \in D) \cdot P(len(H) = n) \cdot \prod_{i=2}^{n} P(w_i|w_1, w_2, \cdot, w_{i-1})$.

Similarly, [Jin and Hauptmann 2001] selected headline words through the NBL, NBF, EM and TF-IDF methods, and then reordered them with a trigram language model. [Zajic et al. 2002] generated headlines for newspaper stories through a Hidden Markov Model. The use of statistic models for learning pruning-rules for parse trees has also been studied in [Knight and Marcu 2000b; Unno et al. 2006].

Compared to rule-based methods, statistics-based methods rely on the availability of training corpus, and are computationally more expensive. However, due to the ability to learn from training data, statistics-based methods are robust and can be extended to different languages and domains, making it possible to generate cross-lingual headlines.

(3) *Summarization-based Methods*

The methods treat headlines as summaries with a very short length, and adapted traditional automatic text summarization techniques to address the headline generation task [Gattani 2007; Goldstein et al. 1999; Marcu 1997; Reimer and Hahn 1988; Salton et al. 1997; Skorokhodko 1972]. As the basic processing unit, salient sentences in the text are ranked for a summary based on certain features, such as term frequency [Luhn 1958], position in text [Edmundson 1969; Erkan and Radev 2004], cue phrases [Erkan and Radev 2004; Paice 1990; Paice and Jones 1993], number of key words or title words in a sentence [Edmundson 1969], and so on. Several machine learning algorithms like Naïve Bayes [Turney 2000], decision trees [Frank et al. 1999], and semi-supervised learning algorithms [Knight and Marcu 2000a, 2002] worked on the features to discover the most salient sentences. Then, single or multiple sentence compression techniques were applied to the salient sentences to generate a final headline. For instance, [Zajic et al. 2004] built a system called Topiary that combines linguistically motivated sentence compression with statistically selected topic terms. [Colmenares et al. 2015] modeled headlines in a feature-rich space and took headline generation as a sequence prediction task using CRF model. [Filippova et al. 2015] compressed the sentence into a headline by deletion with LSTM. [Filippova 2010] built a word graph for multiple sentences and compressed them into a single sentence by finding the shortest paths.

The advantage of the summarization-based headline generation methods is that they treat text summarization and headline generation uniformly as the same task. But resorting to summarization techniques for headline generation may generate headlines of low quality, when the compression ratio is below 10%. Since headlines are typically no more than 15 words, the compression ratio is far less than 10% for many documents like news articles. In addition, adopting summarization techniques is not applicable to cross-lingual headline generation.

2.1.2 Abstractive Headline Generation Methods. Different from extractive methods, where sentences are the basic processing units, abstractive methods select a set of salient phrases, concepts or events as the basic processing unit according to specific principles during candidate extraction.

In abstractive methods, the headline is generated word by word from scratch using sentence synthesis techniques and natural language generation techniques [Tan et al. 2017]. For instance, [Tseng et al. 2006]

mapped the category-specific terms of the news cluster into the common generic term based on the hypernym search algorithm with the help of WordNet, and took the generic term as headline. [Xu et al. 2010] extracted the keywords from the input document using novel word features derived from its relevant Wikipedia articles, then employed the keyword clustering based headline generation procedure to construct a document's headline from the extracted keywords. [Genest and Lapalme 2012] generated a guided summary using handcrafted abstraction schemes, which included rule-based information extraction, heuristic content selection and generation patterns. [Alfonseca et al. 2013] inferred the hidden event from the extracted syntactic patterns of news cluster based on a Noisy-OR Bayesian network, and then replaced the entity placeholders with the observed surface forms to generate the headline. [Sun et al. 2015] extracted the candidate events based on a bipartite graph of lexical chains and events, then obtained a headline by the graph-based multi-sentence compression model.

2.1.3 Neural Headline Generation Methods. Comparing extractive headline generation methods with abstractive ones, extractive approaches can generate more readable headlines, since they tailor human-written sentences to derive the final title. However, as sentences are usually sparse and longer than headlines, their generated headlines are usually less informative. Many times, headlines do not borrow the exact same words as present in the source documents [Gattani 2007].

In contrast, abstractive headline generation methods use phrases, concepts, and events, which are much less sparse, the generated headlines tend to be to-the-point. However, ensuring grammatical correctness and linguistic fluency of the generated headlines based on a set of phrases and concepts is challenging, due to the difficulty and immaturity of natural language generation techniques [Sun et al. 2015].

To overcome the limitation of abstractive methods, recently, neural-based headline generation attracts a lot of attention due to the success of neural sequence-to-sequence (seq2seq) model on machine translation [Bahdanau et al. 2014; Luong et al. 2015a; Sennrich et al. 2016; Sutskever et al. 2014; Wu et al. 2016], text summarization [Chopra et al. 2016; Gu et al. 2016; Lopyrev 2015; Nallapati et al. 2016; Rush et al. 2015], speech recognition [Bahdanau et al. 2016], and video captioning [Venugopalan et al. 2015]. The processing unit of the model is at the document-level. It first encodes the input text into a context vector representation. The context representation in turn constrains the output of the target sequence.

[Kalchbrenner and Blunsom 2013] first applied this model to machine translation, in which the input sentences are mapped into vectors using convolutional neural networks so that the ordering information of the sequence is lost. Later on, [Sutskever et al. 2014] substituted the encoder and the decoder with both LSTM and implemented the first pure neural translation system that outperformed the phrase-based statistic machine translation by a sizeable margin. A potential issue with this encode-decode model is that when compressing the long sequence into a fixed-length vector, the key concepts will be severely lost. To address this issue, [Bahdanau et al. 2014] proposed the attention mechanism which allows the decoder automatically searches for the relevant parts of source sequence as the context.

Many researches have been done to explore different attention mechanisms. [Luong et al. 2015a] examined two classes of attention mechanisms: a global approach that considers all hidden states of the encoder when deriving the context vector, and a local one that chooses to focus only on a small subset of the source positions at a time. The experiment showed that the global attention with dot alignment and the local attention with predictive alignment works best. [Lopyrev 2015] adopted the former best attention

mechanism but implemented it in two different ways: the complex attention which remains unchanged, and the simple attention that split the hidden states into 2 sets to separately compute the attention weight and decode.

In headline generation task, [Rush et al. 2015] firstly employed an encoder-decoder model to generated headline from the lead (first) sentences of news articles. Its encoder is an attention-based convolutional neural network, and the decoder is a feed-forward neural network language model. The model was trained on a large amount of news headlines and selected recapitulative sentences. Following the strategy, [Lopyrev 2015] generated news headlines with both RNN for encode and decoder. To capture the syntactic properties of the sentence, [Tai et al. 2015] proposed a Tree-LSTM that generalizes LSTMs to tree-structured network topologies. Based on Tree-LSTM, [Takase et al. 2016] incorporated the structural syntactic and semantic information into encoders. [Chopra et al. 2016] employed RNN as the encoder, and incorporated the position information of words, which showed significant improvement. Furthermore, many advanced features and mechanisms were proposed to enhance the performance of the encoder-decoder model for text summarization and headline generation. [Nallapati et al. 2016] restricted the decode-vocabulary of each mini-batch to the words in the source documents of that batch and the most frequent words in the target vocabulary to reduce the soft-max penalty. It also captured the keywords using feature rich encoder, used switching generator-pointer to handle the out-of-vocabulary (OOV) problem and applied a word-level encoder and a sentence-level encoder to capture the hierarchical document structure. [Gu et al. 2016] proposed a novel COPYNET model to explore the copying mechanism, which located a certain segment of the input sentence and puts the segment into the output sequence.

Even with LSTM and the attention mechanism, it's still hard for the encoder-encoder model to capture the document/paragraph-level information. To move towards this task, [Li et al. 2015] proposed a hierarchical neural auto-encoder to preserve and reconstruct multi-sentence paragraphs. They used an LSTM model to encode a paragraph into an embedding for sentences and words composing it, then decoded this embedding to reconstruct the original paragraph. [Tan et al. 2017] attempted to generate news headline by encoding different summaries of a news into a summary representation and generating the output with a hierarchical attention mechanism.

2.2 Question Generation

Question generation aims to generate questions from a given sentence or paragraph. One key application of question generation is in the area of education for reading comprehension [Du et al. 2017; Heilman 2011a; Heilman and Smith 2010]. Combining question generation and question answering as dual tasks also enables to improve question answering systems in natural language processing and AI fields [Duan et al. 2016; Tang et al. 2017]. Three typical methods have been developed on question generation, which are *rule-based*, *template-based*, and *neural* [Ali et al. 2010; Chali and Hasan 2015b; Kalady et al. 2010].

2.2.1 Rule-based Question Generation Methods. Rule-based approaches rely on well-designed rules for declarative-to-interrogative sentence transformation based on deep linguistic knowledge [Mitkov and Ha 2003; Rus et al. 2010]. [Heilman and Smith 2010] used manually written rules to generate multiple questions from a sentence, and then ranked the questions through a logistic regression model trained on a tailored dataset consisting of labeled outputs.

2.2.2 Template-based Question Generation Methods. Besides rule-based approaches which exploit syntactic roles of words in generating questions, another group of research turns to manually construct question templates, and then apply them to generate questions [Lindberg et al. 2013; Mazidi and Nielsen 2014; Mostow and Chen 2009]. [Lindberg et al. 2013] introduced a template-based approach which incorporated semantic role labels to generate natural language questions to guide online learning. [Labutov et al. 2015] used crowdsourcing to collect a set of templates for the text and then ranked the relevant templates for the text. It encoded the original text in a low-dimensional ontology, and then aligned the question templates to that space to get the top relevant templates. [Chali and Hasan 2015b] generated questions from a topic, associated with a body of texts containing topic-related useful information. Then, questions are generated by exploiting the named entity information and the predicate argument structures of the sentences present in the body of texts. [Serban et al. 2016] generated simple factoid questions from logic triple (subject, relation, object), where structured representation is mapped to natural language text.

Both rule-based and template-based question generation methods rely on manually generated rules and templates, and the generated questions are constrained by the human-designed transformation which make it hard to scale to other domains.

2.2.3 Neural Question Generation Methods. To overcome the limitations of the above methods, similar to headline generation, currently many researches have shifted to generating questions with the encoder-decoder model. [Serban et al. 2016] proposed a neural network model to generate the factoid questions from FreeBase KB [Bollacker et al. 2008]. Instead of generating from structured triples, [Zhou et al. 2018] generated meaningful and diverse questions from natural language sentences, where the encoder is enriched with answer position and lexical features. [Yuan et al. 2017b] explored the training skill using a combination of supervised and reinforcement learning. Without regards to the answer information, [Du et al. 2017] investigated the effect of encoding sentence- vs. paragraph-level information, of which the sentence-level mode achieved the state-of-the-art performance.

Neural question generation methods use deep sequence-to-sequence learning approach to generate questions. They are fully data-driven and provide an end-to-end solution without the guidance of rules or templates [Du et al. 2017].

3 DATASET AND ANALYSIS

We collect and construct a large-scaled naturally annotated **QA**s dataset, focusing on **Teen**s' commonly-encountered problems.In this section, we first introduce the collection procedure, and then analyze the dataset to show its applicability to our task. Finally, we explore the possibilities to solve other tasks with our dataset.

3.1 Data Collection

The question-answers (QAs) pairs of TeenQA are crawled from Quora[1]. Quora is a popular question-and-answers website created in 2009, where questions are asked, followed, and answered by community users. We choose Quora as data source for the following 2 reasons:

[1]https://www.quora.com/ - a place to share knowledge and better understand the world

- Topic-specific questions. Although knowledge in Quora is open-domain, each asked question is categorized into several topics. Given teens' mostly questioned topics, we collect the relevant QAs pairs.
- High Quality. Not all questions and answers are of high quality, but we can extract the golden set through question's *follows* tag and answer's *upvotes* tag. In general, more *follows* mean the question is paid more attention to. Similarly, more *upvotes* indicate the answer is accepted with good quality.

3.1.1 Question Topic Seeds. According to teens' commonly-encountered stress categories and sub-categories shown in Fig. 1(a), we identify 66 question topics existing in Quora as seeds, each corresponds to a stress sub-category (Table 1). To enrich the dataset, we also crawl the QAs pairs under the related topics of the seeds. Fig. 2 illustrates the crawling process.

Table 1. Question Topic Seeds in TeenQA

Stress Category	Question Topic Seeds
Study	Studying, Study-Habits, Study-Strategies, Colleges-and-Universities, College-and-University-Admissions, Education, Higher-Education, Educational-Courses, Educational-Resources, Competition, Contests-and-Competitions, Grades-education, Academic-Degrees, Homework, Exams-and-Tests, Exam-Strategies, Recommendations, College-Advice, The-College-and-University-Experience, Graduations, Graduate-School-Education, Postgraduate-Education, Scientific-Research, Learning, Studying-Abroad,
Family	Family, Family-Relationships-and-Dynamics, Interpersonal-Interaction-with-Family-Members, Parents, Siblings, Dating-and-Relationships-1, Marriage
Romantic Relation	Affection, Love, Frustration, Dating-Advice, Psychology
Peer Relation	Interpersonal-Interaction, Social-Advice, Friendship, Roommates, Classmates, Social-Psychology
Self-Cognition	Self-Awareness, Self-Motivation, Self-Improvement, Cognitive-Psychology
Employment	Employment, Self-Employment, Jobs-and-Careers, Job-Interviews, Interviews-8, Job-Interview-Questions, Job-Searches, Internships, Part-Time-Jobs, Internship-Hiring, Hiring, Career-Advice, Personal-Finance
Life	Life-and-Living, Tips-and-Hacks-for-Everyday-Life, Life-Lessons, Life-Advice, Motivation, Psychology-of-Everyday-Life

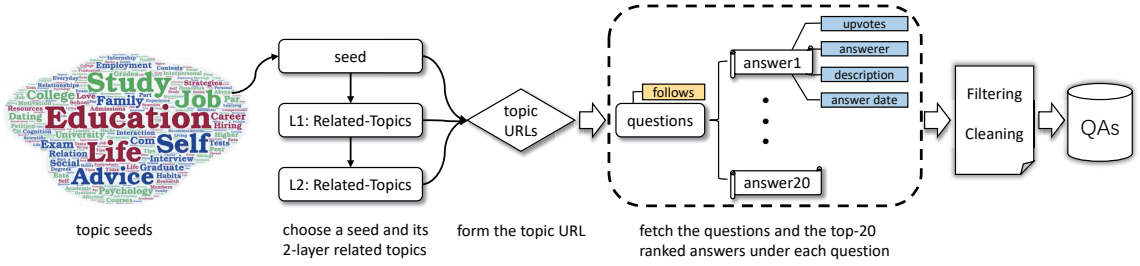


Fig. 2. The crawling process of TeenQA. Quora provides a related-topic-list for each topic, we recursively crawl 2-layer related topics of the topic seed.

3.1.2 Data Filtering and Cleaning. While crawling, we process the data from 3 aspects.

- Question Length. Short questions tend to be more general, consequently harder to derive from answers. On the other hand, longer questions are more specific, but it's not suitable as a theme and may not have enough answers. To balance between generalization and specificity, we keep the questions containing 4 to 15 words.
- Answers Number. For each question, Quora ranks the answers based on answers' *upvotes*, freshness, etc. From Quora's top-20 answers, we further filter out at most top-4 answers based on *upvotes* to ensure collecting high-quality answers for a question.
- Answer Body. As an informal community Q&A site, Quora doesn't constrain users' writing styles. As a result, its answers are colloquial, and contain plenty of unstructured contents, irregular symbols, and white spaces. To clean the data, we only reserve the plain text, normalize the punctuation marks into English style, remove the meaningless lines for separation, and add '.' to sentences if an end punctuation is missing.

In this way, we obtain a QAs dataset *TeenQA*, containing 697,105 questions and 2M answers (2.8 answers per question on average). Table 2 presents an example of QAs pairs in TeenQA, in which the answers are sorted by *upvotes* numbers in descending order.

Table 2. An Example of QAs pairs in TeenQA

Field	**Content**
question	How do top students study?
category	Study
topic	Study-Strategies
follows	19948
answer1	I'll speak on behalf of a close friend of mine, who attended an unknown university from where I am from (Lima, Peru), and got accepted for a fully funded PhD to work with the world-leaders (including Nobel Laureates) at Systems Biology and Computational Biology at Harvard, UC ...
upvotes1	9622
answerer1	Arturo Deza
description1	Robot Opthalmologist
date1	Oct 26, 2016
answer2	I am an above-average student at Caltech. I don't think I study particularly hard, but I do. 1 - Get 8-9 hours of sleep a night. This allows me to go to class well-rested and do my problem sets with greater efficiency. 2 - Always go to class. Even if the lectures are not useful ...
upvotes2	6306
answerer2	Jessica Su
description2	CS PhD student at Stanford
date2	Oct 9, 2012
answer3	What is my story: I went to IIT, one of the best colleges in India, and stood first in my class. Thereafter I did PhD in Chemical Engineering, and worked hard towards my studies. I put a lot of hours towards my studies in my life, and I have seen some other students who were far better...
upvotes3	5491
answerer3	Rohit Malshe
description3	studied at Doctor of Philosophy Degrees
date3	Mar 28, 2017
answer4	I was a decent student back when I was an MIT undergrad and now that I'm a prof, and I really want my students to do well. Every semester, I email them a link to the following article by Cal Newport: http://calnewport.com/blog/2007/..., which pretty much summarizes everything in ...
upvotes4	3740
answerer4	Ben Leong
description4	Associate Professor of Computer Science
date4	May 24, 2012

3.2 Data Analysis

3.2.1 Topics Distribution. Fig. 3 and 4 shows the topic distribution of TeenQA. *study* and *employment* is the biggest two topics, which together take up more than a half of all QAs pairs. The other 5 topics are almost equally sized. The distribution is consistent with teens' mostly concerned doubts and stress categories.

Fig. 3. Topics in TeenQA. The sunburst chart only displays 2 layers of topics due to space limitation.

Topic	#Question	Proportion
Study	220,819	31.7%
Employment	187,949	27.0%
Life	65,312	9.4%
Family	57,328	8.2%
Romantic Relation	56,005	8.0%
Peer Relation	55,407	7.9%
Self-Cognition	54,285	7.8%
Total	**697,105**	100.0%

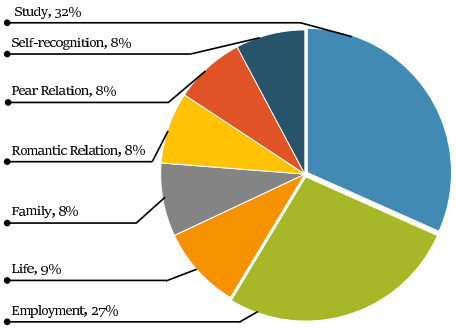


Fig. 4. Topic distribution in TeenQA.

3.2.2 Lengths of Questions and Answers. As shown in Fig. 5, the average length of questions in TeenQA is around 10 words, which is suitable for serving as a theme. The average length of answers, considering all the top-4 answer, is around 165 words. Answers receiving more upvotes usually have more words (information). In the boxplot in Fig. 5, the lines from the bottom to the top denote the minimum, the first quartile, the median, the third quartile, and the maximum of the length. x mark denotes the average value. Dots above the top line denote outliers, whose length could be very big.

Question/Answer	Avg. Length (#Word)
Question	10.2
Answer	165.3
Answer 1	194.6
Answer 2	160.4
Answer 3	145.6
Answer 4	138.3

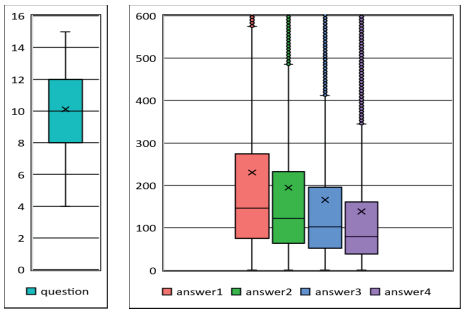


Fig. 5. Average lengths of questions and answers in TeenQA.

3.2.3 Number of Questions' Follows, Questions' Answers, and Answers' Upvotes. Table 3, 4, and 5, respectively, list the numbers of questions' *follow*s, questions' answers, and answers' *upvote*s.

Question's *follows* number can be used to measure users' attention degree to the question and the corresponding topic. In order to know more about teens' stressor events, e-bibliotherapy and education practitioners should pay more concerns to those questions with large *follow*s, especially those with more than 40 follows, which accounts for 4.6% of all.

Table 3. Number of Questions' *follows*

#Follows	#Question	Proportion
[0, 5)	432,256	62.0%
[5, 10)	126,883	18.2%
[10, 20)	70,098	10.1%
[20, 40)	35,786	5.1%
[40, +)	32,082	4.6%
All	**697,105**	100.0%

Table 4. Number of Answers

#Answer	#Question	Proportion
0	22,968	3.3%
1	131,605	18.9%
2	122,359	17.5%
3	94,120	13.5%
4	326,053	46.8%
All	**697,105**	100.0%

Similar to question's *follows* number, answer's *upvotes* number reflects the acceptability and the quality of the answer. Table 5 shows the relation between answers' ranks and answers' *upvote*s, through which we can see that among the total 1,962,895 answers, around 6.2% of them have more than 40 *upvote*s.

3.2.4 Types of Questions. Diversity of question types is an important criterion of dataset's suitability. As the training dataset is for e-bibliotherapy, TeenQA should accommodate different kinds of questions to help generating appropriate reading themes. We extend the categorization of question types made for question answering systems [Mishra and Jain 2016] to consider the following six question types for e-bibliotherapy.

Table 5. Number of Answers' *Upvotes*

#Upvotes	#Answer1	#Answer2	#Answer3	#Answer4	Sum	Proportion
$[0, 5)$	424,473	418,566	342,121	270,178	1,455,338	74.1%
$[5, 10)$	95,827	53,667	33,783	23,574	206,851	10.6%
$[10, 40)$	88,207	43,177	27,403	20,138	178,925	9.1%
$[40, +)$	65,630	27,122	16,866	12,163	121,781	6.2%
All	674,137	542,532	420,173	326,053	**1,962,895**	100.0%

- Factoid questions [what, when, which, who, whom, where]. [2] These questions are simple and fact-based, their answers are a short span of words, entity, or sentence.
- List questions. This type of questions can be decomposed into several factoid questions, e.g., *what are the most popular programming languages?*
- Hypothetical questions [what if, how if]. Hypothetical questions ask for answers based on a hypothesis, which are hard to answer because they are highly subjective to questions and the answers are not specific.
- Confirmation questions [is, will]. This type of questions requires answers in the form of *yes* or *no*. But for this type of questions in Quora, answerers usually provide detailed explanations on why they choose yes or no.
- Causal and Explanatory questions [how, why]. This type of questions asks for answers explaining one phenomenon, which can either be subjective or objective. In order to explain it clearly, the answers tend to be quite long.
- Methodological questions [what should, how can]. This type of questions are highly valuable in our task. When teens encounter problems, they tend to pose questions for detailed solutions. However, this type of questions are the hardest questions to derive from multiple answers. It is because the answers are completely subjective to answerers, and the key information is sparsely spreading across different answers.

Based on the above classification, we randomly sample 1000 questions from TeenQA and analyze their type distribution in Table 6. As we wish, all types of questions are present with considerable proportions and the methodological questions take up the most in TeenQA, accounting for 27.1%.

Table 6. Distribution of six different question types in TeenQA

Question Type	Proportion	Example
Factoid	18.4%	Where can I publish my article for free?
List	11.4%	What are the easiest musical instruments to learn to play well?
Hypothetical	4.1%	What should I do if my phd advisor does not answer my Emails?
Confirmation	19.5%	Is Fedora better than Ubuntu for Software Development?
Causal and Explanatory	19.5%	Why do the most important lessons in life hurt so much to learn?
Methodological	27.1%	How could I get rich as a teenager?

The QAs pairs of TeenQA are strikingly different from the existing QA datasets in the aspect of question diversity, answer complexity, and content subjectivity, which presents a great challenge for our theme generation task from multiple documents.

[2] The interrogatives in [ ] are some examples. In fact, each question type contains far more interrogatives. Similarly hereinafter.

Table 7 lists some recently released popular QA datasets for question generation, question answering, and reading comprehension, where the majority of the questions belong to factoid questions.

Table 7. Recently Released Popular QA Datasets

Dataset	Source	Format	#Doc.	#Q	Question	Answer
SQuAD	Wikipedia	passage-Q-A	536	100K	human generated	span of words
MACRO	Bing	passage-Q-A	1M/ 200K	100K	Bing queries	human generated
TriviaQA	trivia/Bing	documents-Q-A	650K	95K	trivia enthusiast	span of words
WikiQA	Bing	sentences-Q-A	29K	3K	Bing queries	sentence selection
MCtest	children's stories	story-Q-A	660	2640	human generated	multiple choices
NewsQA	CNN news	document-Q-A	10K	100K	human generated	span of words
rc-data	CNN/DM news	document-Q-A	287K	1.3M	cloze	entity
SimpleQuestion	Freebase	subject-relation-object	—	108K	human generated	object
TeenQA	Quora	Q(category,topic,follows)-A(date,upvotes,desc)s	2M	697K	human generated	human generated

- SQuAD [Rajpurkar et al. 2016] collects 100K question-answer pairs from crowdworkers on 536 Wikipedia articles. The answer to each question is a segment of text from the corresponding passage.
- MS MACRO [Nguyen et al. 2016] contains 100K questions, 1M passages and links to over 200K documents. The questions are real queries issued through Bing. The passages are extracted from the web documents returned by Bing, and the answers are human generated based on the related passages.
- TriviaQA [Joshi et al. 2017] includes 95K question-answer pairs from 14 trivia and quiz-league websites, and collects the textual evidence documents from Bing search results and Wikipedia articles. There are on average 6 evidence documents for deriving the answer to a question.
- WikiQA [Yang et al. 2015] contains 3K questions sampled from Bing. Each question is associated with a Wikipedia page based on the user clicks. All sentences of the page summary are extracted as candidates and are labeled on whether the sentence is the correct answer of the questions by crowdsourcing workers. Overall there are 29K sentences obtained.
- MCtext [Richardson et al. 2013] has 660 fictional stories created by crowdworkers and 4 multiple-choice questions per story.
- NewsQA [Trischler et al. 2016] collects 100K question-answer pairs from crowdworkers on 10K news articles from CNN, where the answer is also a span of text from the corresponding articles.
- rc-data [Hermann et al. 2015] includes 287K newspaper articles from CNN/Daily Mail news. Based on this, 1M cloze questions are constructed by replacing the entities with placeholders.
- SimpleQuestions [Bordes et al. 2015] consists of 108K questions written by human English-speaking annotators based on the corresponding facts in Freebase, where a fact is of the form (*subject-relationship-object*), and the answer is *object*.

3.3 More Applicable Scenarios of TeenQA

In addition to the theme generation task, TeenQA could also be applied to other scenarios:

(1) **General Question Generation (GQG) for Reading Comprehension**.The existing QA datasets for reading comprehension (RC) in Table 7 are either for answering the question based on text(s) or for generating questions from sentences or short paragraph(s). However, generating an abstractive question covering the main idea of the whole long text is a more comprehensive measurement for RC, which is far more difficult in that users should understand all textual information first. Furthermore, generating a high-level general question from multi-documents around the same topic is even more tough. To this end, TeenQA is the first large-scaled dataset to support question generation from both single document and multi-documents. TeenQA provides sufficient (general question, answer(s)) pairs, and these answers can be seen as stand-alone documents to derive the general questions.

(2) **Adolescent Stress Analysis**. TeenQA contains nearly 700K topic-specific questions, related to common and typical adolescent stress categories of *study, family, peer relation, romantic relation, self-cognition, employment,* and *life* in general. The attention degree of each question can be inferred from the question's $follows$ number. This makes it possible for the quantitative study of adolescent stress. We hope that TeenQA could help psychologists and education practitioners better understand and solve teens' stress problems.

4 PROBLEM FORMULATION AND SOLUTIONS

4.1 Problem Formulation

Given N recommended articles $\mathbf{RA} = \{\boldsymbol{R_1}, \boldsymbol{R_2}, \cdots, \boldsymbol{R_N}\}$, our task is to generate a theme heading $\mathbf{T} = \{\boldsymbol{t_1}, \boldsymbol{t_2}, \cdots, \boldsymbol{t_K}\}$ in a question form, covering the essential information of $\mathbf{RA}$, $K \leq 16$ is the number of theme tokens, $t_{i<K}$ is an English word, $t_K =$'?' is the end mark of the theme, and $\boldsymbol{t_{<n}} = \{t_1, t_2, ..., t_{n-1}\}$ represents all generated words before t_n. We can describe the task in a probabilistic setting:

$$\hat{\mathbf{T}} = \arg\max_{\mathbf{T}} P(\mathbf{T}|\mathbf{RA}) \tag{1}$$

$$P(\mathbf{T}|\mathbf{RA}) = \prod_{n=1}^{K} P(t_n|\boldsymbol{t_{<n}}, \mathbf{RA}) \tag{2}$$

The task can be viewed as a sequence-to-sequence (seq2seq) task, which can be addressed elegantly by a neural encoder-decoder model [Bahdanau et al. 2014; Sutskever et al. 2014].

4.2 Overview of the Encoder-Decoder Model

For an input sequence $\mathbf{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m\}$, where $\boldsymbol{x}_i$ is the one-hot presentation of words from vocabulary $\mathbf{V}$, the encoder-decoder model first encodes $\mathbf{X}$ into a context vector representation $\boldsymbol{c}$, then decode $\boldsymbol{c}$ to generate the output sequence $\mathbf{Y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_n\}$ word by word based on model parameters $\boldsymbol{\theta}$ and the precedently generated tokens $\boldsymbol{y}_{<i}$.

$$P(\mathbf{Y}|\mathbf{X}; \boldsymbol{\theta}) = \prod_{i=1}^{n} P(\boldsymbol{y}_i|\boldsymbol{y}_{<i}, \mathbf{c}; \boldsymbol{\theta}) \tag{3}$$

$\mathbf{Y}$ with the highest conditional probability is chosen as the final output:

$$\hat{\mathbf{Y}} = \arg\max_{\mathbf{Y}} P(\mathbf{Y}|\mathbf{X};\boldsymbol{\theta}) \tag{4}$$

Encoder. The encoder encodes the input sequence into a context vector representation $\boldsymbol{c}$. There are many variants of encoder, of which Recurrent Neural Network (RNN) [Rumelhart et al. 1986] with LSTM [Hochreiter and Schmidhuber 1997] units is one of the mostly used one owing to its capacity for dealing with long sequence. Before encoding, the input words $\boldsymbol{X}$ are mapped to low-dimensional real-valued embeddings $\boldsymbol{E}$, which carries the semantic information of words:

$$\boldsymbol{E} = \{\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_m\} = \boldsymbol{W}_{emb} \cdot \boldsymbol{X} \tag{5}$$

where $\boldsymbol{W}_{emb} \in \mathbb{R}^{D\times|V|}$ is the embedding matrix, D is the embedding dimension and $|V|$ is the vocabulary size.

Then RNN encoder calculates the hidden state $\boldsymbol{h}_t$ of each word based on the word embedding $\boldsymbol{e}_t$ and the former hidden state $\boldsymbol{h}_{t-1}$:

$$\boldsymbol{h_t} = f(\boldsymbol{e}_t, \boldsymbol{h}_{t-1}) \tag{6}$$

where f indicates the function of RNN unit. We use LSTM unit in our task, so Eq. 6 could be rewritten as:

$$\boldsymbol{h}_t = f(\boldsymbol{e}_t, \boldsymbol{h}_{t-1}) \tag{7}$$

$$= \boldsymbol{o}_t \cdot tanh(\boldsymbol{C}_t) \tag{8}$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_o \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{e}_t] + \boldsymbol{b}_o) \tag{9}$$

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_f \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{e}_t] + \boldsymbol{b}_f) \tag{10}$$

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_i \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{e}_t] + \boldsymbol{b}_i) \tag{11}$$

$$\tilde{\boldsymbol{C}}_t = tanh(\boldsymbol{W}_c \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{e}_t] + \boldsymbol{b}_c) \tag{12}$$

$$\boldsymbol{C}_t = \boldsymbol{f}_t \cdot \boldsymbol{C}_{t-1} + \boldsymbol{i}_t \cdot \tilde{\boldsymbol{C}}_t \tag{13}$$

$$\tag{14}$$

where $\boldsymbol{o}_t$, $\boldsymbol{f}_t$, $\boldsymbol{i}_t$ are output gate, forget gate and input gate, respectively. $\boldsymbol{C}_t$ is the memory cell which aggregates the old memory from forget gate and the new memory from input gate. $\boldsymbol{W}_o$, $\boldsymbol{W}_f$, $\boldsymbol{W}_i$, $\boldsymbol{b}_o$, $\boldsymbol{b}_f$ and $\boldsymbol{b}_i$ are gate parameters.

After all hidden states are obtained, the context vector representation is calculated as:

$$\boldsymbol{c} = g(\{\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_m\}) \tag{15}$$

where g is the context function. Usually we make $g = \boldsymbol{h}_m$ for reason that the last hidden unit can be seen as the compression of $\boldsymbol{X}$.

Decoder. To better capture the sequence information, we also use LSTM-RNN as decoder. It generates the output based on the hidden state $\boldsymbol{s_t}$ and the context vector $\boldsymbol{c}$:

$$\boldsymbol{y}_t = \arg\max_{\boldsymbol{y}} P(\boldsymbol{y}|\boldsymbol{s}_t, \boldsymbol{c};\boldsymbol{\theta}) \tag{16}$$

$$\boldsymbol{s}_t = \psi(\boldsymbol{s}_{t-1}, \boldsymbol{y}_{t-1}) \tag{17}$$

where ψ is the function calculating the current hidden state with respect to the last hidden state and the last output.

Attention. The use of fixed-length context vector makes it hard to cope with long input sequences. [Bahdanau et al. 2014] proposed the attention mechanism which allows the decoder automatically searches for relevant parts of the source sequence as context. Consequently, the calculation of the current output is changed to:

$$\boldsymbol{y}_t = \arg\max_{\boldsymbol{y}} P(\boldsymbol{y}|\boldsymbol{s}_t, \boldsymbol{c}_t; \boldsymbol{\theta}) \tag{18}$$

$$\boldsymbol{c}_t = \sum_{i=1}^{m} \alpha_i^t \boldsymbol{h}_i \tag{19}$$

$$\alpha_i^t = \frac{exp(e_{ti})}{\Sigma_{k=1}^{m} exp(e_{tk})} \tag{20}$$

$$e_{tj} = a(\boldsymbol{s}_t, \boldsymbol{h}_i) \tag{21}$$

where a is the function determining the different attention algorithm.

Training. The encoder-decoder model is mostly trained with maximum likelihood strategy [Ayana et al. 2017]:

$$\mathcal{L}_{MLE}(\boldsymbol{\theta}) = \sum_{(\boldsymbol{X},\boldsymbol{Y})\in\mathcal{D}} logP(\boldsymbol{Y}|\boldsymbol{X};\boldsymbol{\theta}) \tag{22}$$

where $\mathcal{D}$ is the training dataset and $P(\boldsymbol{Y}|\boldsymbol{X};\boldsymbol{\theta})$ is defined in Eq. 3.

Limitations in theme heading generation. Although the encoder-decoder model has demonstrated great success in handling sequence-to-sequence tasks like machine translation, text summarization, speech recognition, and video captioning, it has limitations in theme heading generation for multiple articles.

Firstly, even with the attention mechanism, the encoder-decoder model is still insufficient in coping with long text, because too many textual contents makes the model hard to train and hurts the performance [Cho et al. 2014; Li et al. 2015; Tan et al. 2017].

Secondly, the content feeded into the encoder should be self-consistent so that the context vector makes sense, while the multiple articles in our task may differ a lot from each other, and in some cases contain even diametrically opposed subjective contents and opinions. How to encode these diametrically opposed articles into context representation and extract their general idea in the meanwhile are questions worthy of deep investigation.

We introduce our 4 tries to address the long-text and diverse-content issues in the following subsections.

4.3 Solutions

4.3.1 Solution 1: Encoder-Decoder with Summary on Outputs (ED-SoO). Confronted with multiple articles, our first solution is to generate a theme heading from each individual article, and then pick up the one with the minimum negative log likelihood (NLL) loss as the final theme heading (Fig. 6). Our basic encoder-decoder model ED implements the encoder and decoder with both a 3-layer LSTM-RNN, each layer having 512 hidden units. Like [Lopyrev 2015], a simple yet efficient attention mechanism splits hidden units into 2 parts: the first 472 units for computing the context and decoding words, and the last 40 units for computing

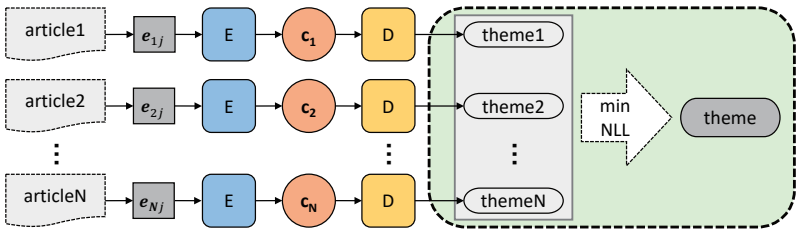


Fig. 6. Solution 1: ED-SoO.

E, D denotes the encoder and decoder, $\boldsymbol{e}_{ij}$ is the word embedding of the j-th word in the i-th article, $\boldsymbol{c}_i$ is the context vector of the i-th article.

the attention weight. In the decoding phrase of training, we use teacher forcing strategy which randomly replace the input word from golden theme with the generated word at last moment with 10% probability.

Table 8. Words overlap between questions (headings) and answers (articles) in TeenQA

words overlap	first 25 words	first 50 words	all words
answer 1	28%	38%	56%
answer 2	27%	37%	53%
answer 3	27%	36%	50%
answer 4	26%	34%	46%

[a] In TeenQA, a question (heading) has 16 words, and an answer (article) has 165 words on average.

To cut down the long text, we perform lead summarization to extract the first m words of each article as input based on our observation that most of the useful information tends to appear in the first small part of the articles (Table 8), where $m \in \{50, 165, 200\}$ in the study.

Solution 1 addresses the diverse-content issue at the heading (rather than article) level. However, it has two problems.

- Low information density. A single article may not have enough information to derive the theme. Take TeenQA for example, the average words overlap between a theme (question) and an article (answer) is around 50% (Table 8).
- Content incompleteness. One article may address user's stress and question from only one perspective, affecting the model to generate a biased theme. Only synthesising all articles could we capture the essential point of the stress and generate a more general theme.

4.3.2 Solution 2: Encoder-Decoder with Summary on Inputs (ED-SoI). To overcome the problems of Solution 1, our second solution is to first synthesize multiple articles with the help of multi-document summarization techniques, and then encode the summary result into an overall context vector, based on which the decoder obtains an output as the theme of these articles.

Multi-document summarization could identify and eliminate the redundancy across articles, recognize novel information in different articles, and make sure the final summary is both coherent and complete. It could be very applicable to deal with the low information density and content incompleteness problems of

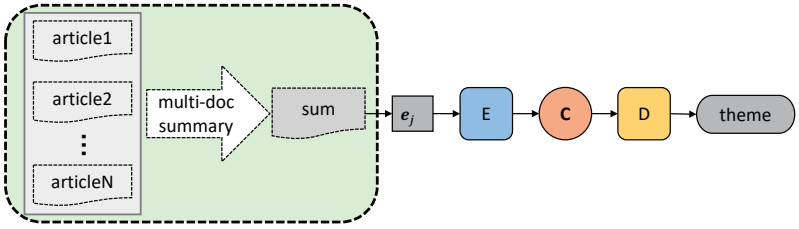


Fig. 7. Solution 2: ED-Sol.

E, D and C denote the encoder, decoder, and context vector, respectively. $\boldsymbol{e}_j$ is the word embedding of the j-th word in the multi-document summary.

Solution 1. In the study, we apply 6 different multi-document summarization techniques to summarize the N input articles into a s-word text, where $s \in \{50, 165, 200\}$.

— Coverage summarization. This is the baseline summarization method. It takes and combines all the lead sentences from each of the articles as the summary of the N input articles.

— Centroid summarization [Radev et al. 2004]. Centroid-based method first constructs a centroid of the articles, which consists of words whose TF-IDF scores are above a pre-defined threshold. The salience of each sentence is computed as the weighted average of the similarity between the sentence and the centroid, sentence position within an article and the similarity between the sentence and the first sentence of the article.

— TextRank summarization [Mihalcea and Tarau 2004]. TextRank is a graph-based sentence ranking model where each sentence is added as a vertex and the sentence similarity (e.g. words overlap) is added as edge between sentences. After the graph-based ranking algorithm converges, we can sort sentences based on their final scores.

— ILP summarization [Gillick and Favre 2009]. Integer Linear Programming (ILP) method takes document summarization task as a combinatorial optimization problem, whose optimization goal is to cover as many word n-gram concepts as possible within the length constraint.

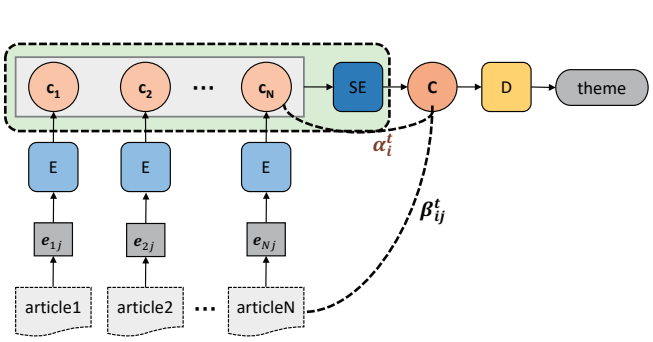
— ClusterCMRW summarization [Wan and Yang 2008]. Cluster-based Conditional Markov Random Walk (ClusterCMRW) method first detects the theme clusters in articles, and then incorporates the cluster-level information and the sentence-to-cluster relationship to compute the saliency score of the sentences based on a conditional markov random walk model.

— Submodular summarization [Lin and Bilmes 2010]. Submodular method formalize the document summarization task as submodular function maximization problem under the budget constraint, where the information coverage, non-redundancy and diversity is reflected in a classes of submodular functions.

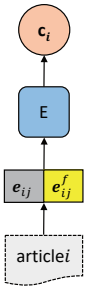
Table 9. Words overlap between questions (headings) and Submodular summaries of 4 answers (articles) in TeenQA

words overlap	**first 25 words**	**first 50 words**	**first 165 words**	**first 200 words**
4 articles' submodular summary	37%	46%	65%	68%

As table 9 shows, multi-document summarization of 4 articles (take Submodular method as example) obtains a significant gain of information density. The content incompleteness could also be solved to a large degree by combining all articles. But there remains a serious problem in multi-document summarization that it assumes the articles of a theme are coherent with each other so as to make sure the summary is meaningful. However, this assumption does not always hold when the answers (articles) are subjective with quite contradictory contents. These non-coherent contents in the same summary will inevitably confuse the model. To avoid this discordance, we further apply an encoder-decoder model with summary on contexts to pick out the general coherent information among all the articles.



(a) Solution 3: ED-SoC.



(b) Solution 4: ED-SoCF. Embedding features into ED-SoC.

Fig. 8. The ED-SoC solution and ED-SoCF solution.

SE denotes the summary encoder that sequentially encodes $\boldsymbol{c}_1$, ..., $\boldsymbol{c}_N$ into an overall context vector C, $\boldsymbol{e}_{ij}$ and $\boldsymbol{e}^f_{ij}$ is the word embedding and feature embedding of the j-th word in the i-th article. α^t_i and β^t_{ij} is the article-level attention weight and the word-level attention weight.

4.3.3 Solution 3: Encoder-Decoder with Summary on Contexts (ED-SoC). Solution 3 independently encodes each article to preserve its content as much as possible, and then summarizes them through a summary encoder SE and a hierarchical attention decoder (Fig 8(a)).

To get rid of the influence of the subjective contents, we take the first m words of each article as input (the same as ED-SoO), through which we can also obtain a higher words overlap between a theme and all input sequences (it reaches 71% in TeenQA). In the encoding phrase, we use the same encoder to encode each article into a vector representation $\boldsymbol{c}_i$. While decoding, the summary encoder SE (which is a single-layer LSTM-RNN) summarizes $\boldsymbol{c}_1$ to $\boldsymbol{c}_N$ into a summary representation $\boldsymbol{C}$, which can be regarded as the context representation of all articles. Then we use the decoder to generate output words based on $\boldsymbol{C}$ and the hierarchical attention mechanism that picks out the appropriate parts across all articles as context. The context vector $\boldsymbol{C}_t$ for generating $\boldsymbol{y}_t$ in Eq. 19 can be recalculated as:

$$\boldsymbol{C}_t = \sum_{i=0}^{N}\sum_{j=1}^{m} \alpha^t_i \beta^t_{ij} \boldsymbol{h}_{ij}, \text{ where } \alpha^t_i = \frac{exp(a(\boldsymbol{s}_t, \boldsymbol{c}_i))}{\Sigma_{k=1}^{N} exp(a(\boldsymbol{s}_t, \boldsymbol{c}_k))} \text{ and } \beta^t_{ij} = \frac{exp(a(\boldsymbol{s}_t, \boldsymbol{h}_{ij})}{\Sigma_{k=1}^{m} exp(a(\boldsymbol{s}_t, \boldsymbol{h}_{ik})} \tag{23}$$

where N is the article number, m is the length of each article, $\boldsymbol{h}_{ij}$ is the hidden state of the j-th word in the i-th article, α^t_i is the article-level attention weight indicating how much attention should be paid on

the i-th article, β_{ij}^t is the word-level attention weight indicating how much attention should be paid on the j-th word of the i-th article.

4.3.4 Solution 4: ED-SoC with Feature-Rich Embeddings (ED-SoCF). We enhance Solution 3 by embedding a set of features, such as topic feature, articles' upvotes feature, TD-IDF feature, as well as two linguistic features (POS a All the features could be concatenated with the embedding of input words.

— Topic. In the task, the generated theme should be highly relevant with users' stress, which could be revealed by the theme topic. As shown in Tab 2, each theme (question) in TeenQA belongs to a topic. We tokenize the topic into tokens, remove the stop words, look up their word embeddings and take their average embedding as the topic feature of each article word.
— Upvotes. Empirically, upvotes number reflects article's quality, and the model should learn to pay more attention to the words from articles of higher quality. Since the articles (answers) in TeenQA are sorted by upvotes, we simply denote the upvotes feature with a 4 dimension one-hot vector that indicate which article the word is from.
— TF-IDF. TF-IDF is a numerical statistic that reflect how important a word is to a document in a collection or corpus. TeenQA is crawled from Quora, adding the TF-IDF feature to each word embedding could highlight the key word and help generating Quora-like questions. To do this, we first calculate the TF-IDF value of each word, then divide the value into 5 buckets with equal size, denoted with a 5 dimension one-hot embedding.
— POS. Parts-of-Speech (POS) indicates the lexical category that a word belongs to. The POS sequence brings in grammatical information for input sentences, with the help of which we can improve the syntax correctness of the generated question. To obtain the POS embedding of the word, we first assign each word with its POS tag using NLTK toolkit[3] and then train a word2vec model implemented in gensim[4] to cast POS tags into 20-dimension embeddings.
— NER. Named Entity Recognition (NER) locates the named entities into the pre-defined categories, which is of great help for determining the correct interrogative of the question. We use NLTK toolkit to find six types of named entity: PERSON, ORGANIZATION, LOCATION, GPE, FACILITY, GSP, and train them into 10-dimension embeddings using word2vec model as well.

We concatenate the word embedding with the features embedding (Fig 8(b)):

$$\tilde{\boldsymbol{e}_{ij}} = [\boldsymbol{e}_{ij}, \boldsymbol{e}_{ij}^f] \tag{24}$$

where $\boldsymbol{e}_{ij}$ is the original embedding of the j-th word in the i-th article, and $\boldsymbol{e}_{ij}^f$ is the best combination of topic, upvotes, TF-IDF, POS, and NER embeddings.

5 PERFORMANCE STUDY

We conduct four experiments to examine the performance of our four question-like theme heading generation solutions. The experimental setup (including dataset used, implementation details, and performance metrics) and experimental results are reported in the section.

[3] http://www.nltk.org/
[4] https://radimrehurek.com/gensim/models/word2vec.html

5.1 Experimental Set-up

5.1.1 **Dataset**. We use TeenQA as our dataset $\mathcal{D}$. As Table 3 shows, $\mathcal{D}$ is comprised of 326,053 (question, 4 answers) pairs, where each question can be viewed as a theme, and its paired 4 answers can be viewed as the articles under the theme.

In the data cleanup phase, we 1) delete the leading digits or symbols of each list item in articles. For each obtained single/multi-document summary, we 2) lowercase the summary and the theme heading, 3) remove line breaks, 4) tokenize the text with NLTK, 5) delete all symbols except the single quotation mark, and 6) replace digits with '#'. Finally, we split $\mathcal{D}$ into 3 parts: 324,053 pairs for training, 1000 pairs for validating, and 1000 pairs for testing.

5.1.2 **Implementation Details**.

- Word Embedding Initialization. We keep the 40,000 most frequent words as vocabulary V and replace the others with $\langle unk \rangle$ symbols. The token $\langle eos \rangle$ indicating the end of the sequence is padded to both the theme and the article. The words in vocabulary are casted to 100-dimension embeddings using GloVe[5], the embeddings of the other words of low-frequency are randomly initialized with the same scale as GloVe. The word embedding as well as the feature embedding is further trained while learning.
- Multi-document Summarization Methods. The multi-document summarization methods introduced in section 4.3.2 are implemented by [Zhang et al. 2016][6], which is a Java toolkit supporting different summarization tasks and methods as well as integrating Stanford Tokenizer[7] and PorterStemmer [8] for text processing. We also remove the stopwords of answers based on the provided stopword list.
- Training Setup. Our basic ED model is implemented using Keras[9] with Theano[10] as backend. We use the Adam [Kingma and Ba 2014] optimizer with default parameters in training and beam size of 10 in decoding. Batch size is set to 128 and each epoch processes 30080 examples. The model is trained on a single TITAN Xp GPU. Convergence is reached at about 80 to 150 epoches as the input length varies from 50 to 200.

5.1.3 **Evaluation Metrics**. We compare the generated question-like theme headings with the ones posed by humans in TeenQA. Due to the statement flexibility of questions (i.e., the same question could be asked in different ways), we incorporate three types of metrics to measure the lexical similarity, consensus similarity, and semantic similarity of the generated question-like theme headings with respect to human's.

Lexical Similarity Metrics. The computation of lexical similarity is based on the word-overlap between generated themes and the referenced ones. We consider the following three lexical similarity metrics.

(L-B$_n$) **B**LEU [Papineni et al. 2002] is a widely used precision-based metric in machine translation. It analyzes the co-occurrences of n-grams between the generated theme and the reference one. n-gram represents the word group with n words, BLEU-n score is the weighted geometric mean of the individual n-gram precision, where $n = 1, 2, 3, 4$.

[5]http://nlp.stanford.edu/projects/glove/
[6]https://github.com/PKULCWM/PKUSUMSUM
[7]http://nlp.stanford.edu/software/tokenizer.html
[8]http://tartarus.org/ martin/PorterStemmer/
[9]https://github.com/keras-team/keras
[10]https://github.com/Theano/Theano

(L-R_L) **R**OUGE [Lin 2004] is a popular recall-based metric in text summarization. We evaluate the generated themes using ROUGE-L, which is the F-measure of the longest common subsequence (LCS) between the generated theme and the reference one.

(L-M) **M**ETEOR [Denkowski and Lavie 2014] is a metric based on the harmonic mean between unigram precision and recall, which correlates better at the sentence level with human evaluation [Sharma et al. 2017].

Consensus Metric.

(C-Cr) **C**IDE**r** [Vedantam et al. 2015] metric is firstly proposed to measure the human consensus of image captions. It computes the TF-IDF weighting for each n-gram, and computes the CIDEr score by averaging the cosine similarity of the TF-IDF vectors of two sentences. We apply this metric to evaluate how human-like the generated themes are with the human written ones.

Semantic Similarity Metrics. This type of metrics compute the semantic similarity with the help of word embeddings. Like [Vedantam et al. 2015], we first obtain the sentence embedding from embeddings of words making up this sentence, then calculate the cosine similarity between sentences' embeddings as the semantic similarity. These metric scores remain high even though the generated theme is stated in a different way but expresses the similar meaning, while the lexical metric scores will drop markedly.

(S-ST) **S**kip-**T**hought Cosine Similarity. The Skip-Thought model [Kiros et al. 2015] uses an RNN to encode the sentence into an embedding, which has a robust performance on semantic relatedness task.

(S-EA) **E**mbedding **A**verage Cosine Similarity. The embedding of a sentence is computed by averaging the embedding of each word of it.

(S-VE) **V**ector **E**xtrema Cosine Similarity. Vector Extrema [Forgues et al. 2014] computes the sentence-level embedding by extracting the extrema value of each dimension of the embeddings of the words composing this sentence.

(S-GM) **G**reedy **M**atching Score. Greedy Matching [Rus and Lintean 2012] first greedily matches the word of one sentence to the word of another sentence based on the cosine similarity of their words' embeddings, then computes the sentence similarity by averaging these similarities.

The three types of metrics are computed using the Python toolkit published by [Sharma et al. 2017][11], in which the semantic metrics make use of the 300-dimension word embeddings from GloVe.

5.2 Experiments and Results

5.2.1 Experiment 1: Performance Comparison of Different Solutions. We perform three tests in Experiment 1.

Test 1. To investigate the feasibility of the underlying encoder-decoder model and the impact of diverse-content issue of multiple articles, we apply our basic encoder-decoder model *ED* and the state-of-the-art encoder-decoder model *ngq* to generate a question-like theme heading from a single article (like the previous question generation task):

[11] https://github.com/Maluuba/nlg-eval

- **ED-avg**. We train *ED* on every article of training examples. While testing, we generate a theme separately from each article of testing examples and take their **average** scores on evaluation metrics as the final score. Article's lead summary is truncated to be of length $m = 50$.
- **ngq-avg**. It is the same as ED-avg except that we substitute our encoder-decoder *ED* with *nqg* model. *nqg* [Du et al. 2017] is the state-of-the-art model for question generation on SQuAD dataset, both its encoder and decoder are 2-layer LSTM RNNs with 600 hidden units while the RNN of the encoder is bidirectional. *nqg* implements two variations of encoders: one that only encodes the sentence and the other encodes both sentence and paragraph-level information, the first one achieves the best performance. *nqg* generates the output with global attention mechanism [Luong et al. 2015b]. We use the better *nqg* model with only sentence-level encoder.

Test 2. We apply our four solutions (ED-SoC, ED-SoI, ED-SoC, ED-SoCF) to generate question-like theme headings from multiple N articles (N=4), where the summary length of a single article is $m = 50$ (for ED-SoO, ED-SoC and ED-SoCF), the coverage summary length of multiple articles is $s = 165$ (for ED-SoI), and the features embedded into ED-SoCF are POS, TF-IDF, Upvotes and NER.

Test 3. We also compare the machine's behaviors with those of human users by inviting six volunteers (who are all graduate students, including 2 native English speakers and 4 non-native English speakers). 107 (question, answers) pairs are randomly selected from the testing set, and divided equally into 6 sets, one set per volunteer, who is responsible for giving theme headings for multiple articles.

Table 10. Performance Comparison of Different Solutions.

Model	lexical similarity						consensus	semantic similarity			
	L-B$_1$	**L-B**$_2$	**L-B**$_3$	**L-B**$_4$	**L-R**$_L$	**L-M**	**C-Cr**	**S-ST**	**S-EA**	**S-VE**	**S-GM**
ED-avg	0.236	0.136	0.088	0.060	0.234	0.117	0.593	0.472	0.842	0.494	0.664
nqg-avg	0.239	0.149	0.103	0.075	0.242	0.121	0.681	0.477	0.836	0.495	0.667
ED-SoO	0.244	0.147	0.099	0.071	0.252	0.125	0.742	0.482	0.841	0.503	0.671
ED-SoI	0.274	0.160	0.102	0.069	0.272	0.132	0.716	0.491	0.858	0.524	0.686
ED-SoC	0.287	0.168	0.108	0.073	0.278	0.140	0.794	0.488	0.863	0.538	0.698
ED-SoCF	**0.323**	**0.204**	**0.142**	**0.103**	**0.320**	**0.161**	**1.095**	**0.508**	**0.872**	**0.557**	**0.713**
↑	*35.1%*	*36.9%*	*37.9%*	*37.3%*	*32.2%*	*33.1%*	*60.8%*	*6.5%*	*4.3%*	*12.5%*	*6.9%*
Volunteers	*0.406*	*0.282*	*0.198*	*0.139*	*0.398*	*0.218*	*1.809*	*0.594*	*0.877*	*0.658*	*0.765*

[a] The metrics from left to right are **B**LEU-**1**, **B**LEU-**2**, **B**LEU-**3**, **B**LEU-**4**, **R**OUGE-**L**, **M**ETEOR, **CIDEr**, **S**kip-**T**hought Cosine Similarity, **E**mbedding **A**verage Cosine Similarity, **V**ector **E**xtrema Cosine Similarity and **G**reedy **M**atching Score.
[b] The highest score of our solutions on each metric marked in **bold**. The volunteers' scores are shown in *italics*.
[c] ↑ indicates the performance gains of the best solution ED-SoCF compared to the better contrast test nqg-avg.

The results are shown in Table 10, from which we can get the following interesting observations:

— Our task is very difficult for humans (volunteers). Different from previous text summarization task or question generation task where the humans can achieve very high metric scores, the human performance in our task is not that satisfactory. This is because on the one hand, the key information of the theme may be deep hidden in articles, which requires much reasoning skills to derive. On the other hand, it is very hard to express the question-like theme in the same form as the reference one even though humans obtain the key information of the articles.

— ED-avg and nqg-avg performs poor in our question-like theme heading generation task, especially in the consensus metric and the semantic similarity metrics. Although both of them are feasible to

generate fair fluent questions, their content terribly digresses from the general idea, which reflects the limitations of the single encoder-decoder in handling the diverse-content articles. Among the two methods, ED-avg is worse than nqg-avg, because ED-avg uses the most basic encoder-decoder *ED* in which the word embedding dimension and RNN size is much smaller, the encoder is unidirectional, and the optimizer is set to adam with the default parameters without intensively study. Even with this weak encoder-decoder, our solutions based on *ED* in Test 2 still beat nqg-avg by a large margin, demonstrating the effectiveness of our solutions.

— ED-SoO, ED-SoI and ED-SoC comprehensively improves the theme quality but contributes most to the consensus metric and the semantic similarity metrics. It reveals the ability of our solutions to capture the key information across articles. Among the three solutions, ED-SoC achieves the best performance, indicating that summarization on articles's vector representations with hierarchical attention is the best synthesizing method for multiple articles.

— Features can greatly enhance the performance on all metrics. After adding POS, TF-IDF, Upvotes and NER features, ED-SoCF model increases the lexical scores and consensus score by more than 30% and 60% respectively compared to nqg-avg. Some semantic scores (e.g. Embedding Average Cosine Similarity) is comparable with humans (volunteers).

An Example. Table 11 provides an example for concrete study of the performance of each solution. To get an idea of the difficulty of our task, we suggest readers try to derive the theme by hand at first.

Each article seems like talking about the different thing, but after reading all articles, humans can conclude that the theme should be of methodological type and the key information is *spiritual journey*. It should be noted that, the volunteer generated theme begins with *how to* while the reference one begins with *what is the best way*, suchlike differences in expression habits can explain the unsatisfactory performance of volunteers in one aspect.

Looking at the results of the basic *ED* model. It generates one theme respectively from each article. Without considering the overall information, the generated questions are all biased towards the individuality of each article, some of them are far away from the main idea.

ED-SoO picks the best one (the second one) from the themes generated by *ED*. It is much better but the word "monk" is not that suitable.

As for ED-SoI, it abstracts out the phrase *develop a spiritual*, which is not present in articles but fairly close to the main idea. However the question words *how long* is not that suitable.

Finally let's look at the theme generated by ED-SoC and ED-SoCF. ED-SoC captures the correct question words *what is the best way*. And further on, with the help of features, ED-SoCF generates a more complete sentence and conveys almost the same main idea with the reference one.

Table 11. An example for concrete study[12]

Reference	*what is the best way to start a journey in spirituality?*
Volunteers	how **to start a spiritual journey**?
ED	what is the point of living in india?
	how can i become a spiritual monk?
	is i too old to start learning martial arts?

[12]In the generated questions, we highlight the parts appearing in the reference question in **bold**.

	what is the best way to spiritual enlightenment?
ED-SoO	how can i become **a spiritual** monk?
ED-SoI	how long does it take **to** develop **a spiritual**?
ED-SoC	**what is the best way to** become **a spiritual**?
ED-SoCF	**what is the best way to start a spiritual** awakening?
answer1	The truth is that the destination is here right now, yet we always think in terms of a journey. This means, if we think a journey is involved, then we put the goal or destination far away from us. Let me give a small tip based off of a lifetime of practice, of many ups and downs, a lot of study, a lot of meditation - traveling all over the planet: Sit down when you have a half an hour. Don't be in a rush. Bring your awareness to your heart. Make the thought, My heart is my destination. I am with my Goal. Allow yourself to feel this for some time. Always start your practice, whatever it will be, with this thought that you are already there. This will set the tone for your journey-less journey. This is the correct attitude to adopt with great confidence - because it is the truth. May your arrival precede your departure. Love to All, Brian.
answer2	Become part of Spiritual organisation like ISKCON, attend their weekly programmes on various spiritual topics. Start reading Spiritual books written by enlightened masters like Srila Prabhupada, spend time with spiritual people -The ISKCON devotees, do more of spiritual activities like chanting, have Deity Darshan, honour Krishna Prasadam, attend festivals, follow a spiritual lifestyle and you would have begun your spiritual journey. Visit nearest ISKCON centre to know more. Hare Krishna.
answer3	I would suggest that, go and do the entry level course of THE ART OF LIVING, which is YES+! (youth empowerment and skill workshop) for age group of 18-35. or HAPPINESS PROGRAM (aka WAVES OF HAPPINESS) for age group of 18+, for your spiritual journey you'll learn an excellent technique called SUDARSHAN KRIYA , which will harmonies your body and soul with the nature. also along with it, you'll get some knowledge point in the course, which will be beneficial to you for living happy life in this materialistic world. hope this will help you. stay happy stay blessed.
answer4	Asking this question tells me you are already on your spiritual journey. Spiritual energies now are very strong and many people are responding to higher energy frequencies. Asking for the best way makes a lot of sense. One can walk around the mountain for several life times before reaching the higher spheres or one can do the same in one life time going up straight step by step. Master ⟨*anonymity*⟩[13], in his book ⟨*anonymity*⟩ has given mankind such a ladder to heaven, whereby each step brings one to a higher level working at the same time on a healthier body while fostering a healthier and more elevated mind. The method used consists of slow meditative/qigong movements that energize the body with energies of the universe and unblock any energy blockages. The most important part is the study of the laws of the universe and how we as humans are part of these laws, how we can interact with them in a positive or in a negative way. Doing our best trying to live with these laws is what we call cultivation and this is what makes the ⟨*anonymity*⟩ practice a cultivation of mind and body. Reading the ⟨*anonymity*⟩ book from beginning to end without too much interruption is the best way I know not only to start but it will put you on your way to successfully end your spiritual journey.

5.2.2 Experiment 2: Impact of Different Features in ED-SoCF. For the best ED-SoCF solution, we look into the performance gains of its different features. We conduct 5 tests, of which the number of features varies from 1 to 5. In each test, we find the feature(s) with the biggest performance gains, then based on that we add another new feature in next test. Repeating this process until all features are tested.

From Table 12, we can see that:

— Any of the features could improve the performance, of which Upvotes mainly contributes to semantic similarity metrics; TF-IDF mainly contributes to lexical similarity metrics (especially BLEU of long grams) and consensus metric; Topic, NER and POS contributes to all metrics, but individually speaking, POS is the best feature.

— The optimal combination of features is [POS, TF-IDF, Upvotes, NER]. To our surprise, the combination of best single features (e.g. [POS, NER]) does not necessarily get a better result, and integrating

[13] We hide the real name with respect to political issues, without affecting understanding of the main idea.

Table 12. Influence of features on ED-SoCF

	lexical similarity						consensus	semantic similarity			
	L-B$_1$	**L-B$_2$**	**L-B$_3$**	**L-B$_4$**	**L-R$_L$**	**L-M**	**C-Cr**	**S-ST**	**S-EA**	**S-VE**	**S-GM**
ED-SoC	0.287	0.168	0.108	0.073	0.278	0.140	0.794	0.488	0.863	0.538	0.698
Feature(s)											
+U (upvotes)	0.293	0.170	0.108	0.073	0.283	0.141	0.798	0.490	0.865	0.539	0.698
+TI (tf-idf)	0.289	0.175	0.116	0.080	0.285	0.140	0.815	0.489	0.862	0.531	0.695
+T (topic)	0.299	0.176	0.115	0.079	0.289	0.144	0.840	0.489	0.865	0.540	0.701
+N (ner)	0.304	0.183	0.122	0.084	0.295	0.146	0.877	0.498	0.867	0.545	0.703
+P (pos)∗	0.309	0.187	0.128	0.092	0.303	0.152	0.957	0.502	0.867	0.548	0.706
+[P,U]	0.303	0.182	0.122	0.085	0.293	0.146	0.876	0.498	0.866	0.539	0.701
+[P,T]	0.302	0.185	0.124	0.087	0.294	0.146	0.896	0.498	0.864	0.535	0.697
+[P,N]	0.306	0.188	0.126	0.089	0.299	0.149	0.929	0.495	0.866	0.540	0.702
+[P,TI]∗	0.318	0.201	0.139	0.101	0.315	0.154	1.050	0.505	0.869	0.551	0.709
+[P,TI,T]	0.296	0.181	0.123	0.088	0.287	0.143	0.889	0.492	0.864	0.537	0.697
+[P,TI,N]	0.309	0.190	0.129	0.091	0.301	0.148	0.919	0.498	0.865	0.538	0.701
+[P,TI,U]∗	0.312	0.192	0.129	0.089	0.304	0.152	0.937	0.501	0.869	0.548	0.704
+[P,TI,U,T]	0.315	0.199	0.137	0.099	0.308	0.152	0.010	0.506	0.867	0.544	0.705
+[P,TI,U,N]∗	**0.323**	**0.204**	**0.142**	**0.103**	**0.320**	**0.161**	**1.095**	**0.508**	**0.872**	**0.557**	**0.713**
+[P,TI,U,N,T]∗	0.299	0.181	0.123	0.087	0.293	0.143	0.885	0.493	0.864	0.538	0.698

[a] The best combination of feature(s) in each group is marked with ∗.
[b] The highest score on each metric is marked in **bold**.

all features obtains a performance even worse than the single feature. The phenomenon may be caused by the performance offset of the features with similar effects.

5.2.3 Experiment 3: Impact of Articles Summary Lengths. Experiment 3 aims to find the best length m of article's lead summary, and the best length s of articles' multi-document summary. Without impact on comparison result, when testing m we train our basic encoder-decoder model *ED* only on the first article of each training example and vary the length of the lead summary to 50, 165 and 200. Without loss of generality, when testing s we take Submodular method as example to analyze the impact of different summary length (50, 165, 200) on performance of ED-SoI model. The result is shown in Table 13.

Table 13. Influence of summary length

Model	lexical similarity						consensus	semantic similarity			
	L-B$_1$	**L-B$_2$**	**L-B$_3$**	**L-B$_4$**	**L-R$_L$**	**L-M**	**C-Cr**	**S-ST**	**S-EA**	**S-VE**	**S-GM**
ED-50	0.211	0.112	0.069	0.045	0.207	0.100	0.417	**0.454**	0.832	0.466	0.649
ED-165	**0.215**	**0.116**	**0.074**	**0.052**	**0.211**	**0.103**	**0.470**	0.453	**0.837**	**0.473**	**0.653**
ED-200	0.188	0.098	0.061	0.040	0.187	0.086	0.306	0.438	0.820	0.432	0.628
ED-SoI-50	0.230	0.126	0.077	0.049	0.224	0.110	0.560	0.468	0.844	0.496	0.661
ED-SoI-165	**0.253**	**0.143**	**0.092**	**0.062**	**0.248**	**0.123**	**0.643**	**0.484**	**0.850**	**0.509**	**0.676**
ED-SoI-200	0.227	0.125	0.079	0.052	0.220	0.110	0.519	0.469	0.840	0.482	0.657

[a] The highest score on each metric in each group is marked in **bold**.

It is observed that the information quantity and density of the input significantly influence the performance. Increasing the summary length from 50 to 165, the information quantity is improved, the model performance is also improved at the same time. But if continuing increasing the summary length to 200

words, the performance dramatically drops because the information density lowers down and too much subjective information confuses the model and hurt the performance. So we fix the length of multi-document summary as 165 words. In the meanwhile, we note that increasing the input length of *ED* from 50 to 165 does not gain much benefits but costs much longer time to convergent. So we still use the first 50 words of article's lead summary.

Finally, we set $m = 50$ and $s = 165$.

5.2.4 Experiment 4: Impact of Multi-Document Summarization Methods in ED-SoI. Experiment 4 tests the effect of different multi-document summary on model performance. Based on the result of last experiment, we fix the multi-document summary to be of the best length $s = 165$ and change the summarization method to be Coverage, Centroid, Textrank, ILP, ClusterCMRW and Submodular.

Table 14. Influence of multi-document summarization method on ED-SoI

Summary	lexical similarity						consensus	semantic similarity			
	L-B$_1$	**L-B$_2$**	**L-B$_3$**	**L-B$_4$**	**L-R$_L$**	**L-M**	**C-Cr**	**S-ST**	**S-EA**	**S-VE**	**S-GM**
centroid	0.220	0.118	0.074	0.049	0.213	0.104	0.473	0.462	0.840	0.481	0.656
ilp	0.222	0.122	0.076	0.049	0.218	0.107	0.493	0.465	0.841	0.485	0.658
textrank	0.225	0.119	0.072	0.047	0.219	0.110	0.540	0.472	0.844	0.495	0.662
clustercmrw	0.252	0.137	0.085	0.054	0.243	0.124	0.644	0.484	0.851	0.518	0.677
submodular	0.253	0.143	0.092	0.062	0.248	0.123	0.643	0.484	0.850	0.509	0.676
coverage	**0.274**	**0.160**	**0.102**	**0.069**	**0.272**	**0.132**	**0.716**	**0.491**	**0.858**	**0.524**	**0.686**

[a] The highest score on each metric is marked in **bold**.

As Table 14 shows, multi-document summary can greatly enhance the model, but the performance of different summarization methods differs a lot. Among them, the baseline method *Coverage* performs best. It proves our assumption that the generic information mostly spreads in the lead sentences. Oppositely, *Centriod*, *TextRank* and *ILP* which take into account all sentences in articles are affected by the subjective information and perform even poorer than the basic solution *ED-avg*. *Cluster* is better than the above 3 methods for reason that it collects sentences based on detected themes. *Submodular* is the best multi-summarization method in [Zhang et al. 2016] in DUC 2004 task[14], but in our task it falls behind the coverage method by a large margin, indicating the difference of articles between TeenQA and DUC 2004 (newspaper news).

6 CONCLUSION

In order to save users' reading time and provide better stress easing effect in e-bibliotherapy, we propose to generate a theme heading in the form of question for the recommended multiple articles in this work. This task is very different from and much harder than the previous question generation tasks in that we generate questions from multiple long articles and the question could be of any type. We present four task solutions based on the neural encoder-decoder model: encoder-decoder with summary on outputs (ED-SoO), encoder-decoder with summary on inputs (ED-SoI) and encoder-decoder with summary on contexts with(out) feature-rich embeddings (ED-SoC and ED-SoCF). To train the model, we collect and build a large-scale topic-specific question and answers dataset TeenQA, which contains teens' common stressful problems

[14] http://duc.nist.gov/duc2004

and community-based solutions. Due to the diversity and flexibility of questions, we adopt 3 groups of metrics (lexical similarity, human consensus, and semantic similarity) to comprehensively investigate the performance of the solutions. The experimental results show that the ED-SoCF solution overwhelms the state-of-the-art question generation model *nqg* by more than 30% on lexical similarity metrics and 60% on human consensus metric. In the meanwhile the performance on semantic similarity metrics is comparable to humans. It demonstrates the ability of our solutions on capturing the main idea of the articles and synthesizing the key points of each article into a proper question.

Besides a necessary question-like theme heading for multiple recommended articles, to enhance the effectiveness of reading and e-bibliotherapy, multi-articles abstractive summary, as well as a series of prior-and-after reading questions, are also desirable for further research.

REFERENCES

Enrique Alfonseca, Daniele Pighin, and Guillermo Garrido. 2013. HEADY: News headline abstraction through event pattern clustering. In *Proc. of ACL*. 1243–1253.

Husam Ali, Yllias Chali, and Sadid A Hasan. 2010. Automation of question generation from sentences. In *Proc. of the Third Workshop on Question Generation (QG)*. 58–67.

Ayana, Shi-Qi Shen, Yan-Kai Lin, Cun-Chao Tu, Yu Zhao, Zhiyuan Liu, and Mao-Song Sun. 2017. Recent Advances on Neural Headline Generation. 32 (07 2017), 768–784.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio. 2016. End-to-end attention-based large vocabulary speech recognition. In *Proc. of the IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*. 4945–4949.

Michele Banko, Vibhu O Mittal, and Michael J Witbrock. 2000. Headline generation based on statistical translation. In *Proc. of the 38th Annual Meeting on Association for Computational Linguistics*. 318–325.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proc. of the 2008 ACM SIGMOD international conference on Management of data*. 1247–1250.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075* (2015).

David D. Burns. 1980. *Feeling Good: The New Mood Therapy*. William Morrow and Company.

Yllias Chali and Sina Golestanirad. 2016. Ranking automatically generated questions using common human queries. In *Proc. of Intl. Conf. on Natural Language Generation*. 217C221.

Y. Chali and S.A. Hasan. 2015a. Towards topic-to-question generation. *Computational Linguistics* 41, 1 (February 2015), 1–20.

Y. Chali and S.A. Hasan. 2015b. Towards topic-to-question generation. *Computational Linguistics* 41, 1 (February 2015), 1–20.

Kyunghyun Cho, B van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. *On the properties of neural machine translation: Encoder-decoder approaches*.

Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proc. of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 93–98.

Carlos A Colmenares, Marina Litvak, Amin Mantrach, and Fabrizio Silvestri. 2015. HEADS: Headline Generation as Sequence Prediction Using an Abstract Feature-Rich Space.. In *Proc. of HLT-NAACL*. 133–142.

Samuel McChord Crothers. 1917. *A literary clinic*. Houghton Mifflin.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proc. of the ninth workshop on statistical machine translation*. 376–380.

Bonnie Dorr, David Zajic, and Richard Schwartz. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *Proc. of the HLT-NAACL 03 on Text summarization workshop*. 1–8.

X. Du, J. Shao, and C. Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proc. of the 55th Annual Meeting of the Association for Computational Linguistics*. 1342–1352.

N. Duan, D. Tang, P. Chen, and M. Zhou. 2016. Question Generation for Question Answering. (2016), 877–885.
H.P. Edmundson. 1969. New methods in automatic extracting. *Journal of Association for Computing Machinery* 16 (1969).
Peli-Hasz Elizabeth, Szilagyi Simon MD, and Baranyai Szilvia Ma. 2016. E-Bibliotherapy, Computer Based Bibliotherapy - Development Perspectives in Relation to The Effectiveness, Reliability and Economy. In *Proc. of 55th ISERD International Conference*. Houston, USA, 51–56.
G. Erkan and D.R. Radev. 2004. LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. *Journal of Artificial Intelligence Research* 22 (2004), 457–479.
Katja Filippova. 2010. Multi-sentence compression: Finding shortest paths in word graphs. In *Proc. of the 23rd International Conference on Computational Linguistics*. 322–330.
Katja Filippova, Enrique Alfonseca, Carlos A Colmenares, Lukasz Kaiser, and Oriol Vinyals. 2015. Sentence Compression by Deletion with LSTMs.. In *Proc. of EMNLP*. 360–368.
Gabriel Forgues, Joelle Pineau, Jean-Marie Larchevêque, and Réal Tremblay. 2014. Bootstrapping dialog systems with word embeddings. In *NIPS, Modern Machine Learning and Natural Language Processing Workshop*.
E. Frank, G.W. Paynter, I.H. Witten, C. Gutwin, and C.G. Nevill-Manning. 1999. Domain-specific keyphrase extraction. In *Proc. of IJCAI*. 668–673.
A.K. Gattani. 2007. *Automated natural language headline generation using discriminative machine learning models*. Master's thesis. Simon Fraser University.
Pierre-Etienne Genest and Guy Lapalme. 2012. Fully abstractive approach to guided summarization. In *Proc. of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*. 354–358.
Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *Proc. of the Workshop on Integer Linear Programming for Natural Langauge Processing*. 10–18.
J. Goldstein, M. Kantrowitz, V.O. Mittal, and J.G. Carbonell. 1999. Summarizing text documents: sentence selection and evaluation metrics. In *Proc. of ACM SIGIR*.
Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics*. 1631–1640.
Michael Heilman. 2011a. *Automatic factual question generation from text*. Ph.D. Dissertation. Carnegie Mellon University.
Michael Heilman. 2011b. *Automatic Factual Question Generation from Text*. Ph.D. Dissertation. Pittsburgh, PA, USA. Advisor(s) Smith, Noah A. AAI3528179.
Michael Heilman and Noah A Smith. 2010. Good question! statistical ranking for question generation. In *Proc. of the 2010 Annual Conf. of the North American Chapter of the Association for Computational Linguistics*. 609–617.
Karl Moritz Hermann, Tomáš Kočciský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching Machines to Read and Comprehend. In *Proc. of Intl. Conf. on Advances in Neural Information Processing Systems*. 1693–1701.
Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780.
Rong Jin and Alexander G Hauptmann. 2001. Automatic title generation for spoken broadcast news. In *Proc. of the first international conference on Human language technology research*. 1–3.
Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Proc. of the 55th Annual Meeting of the Association for Computational Linguistics*. 1601–1611.
Saidalavi Kalady, Ajeesh Elikkottil, and Rajarshi Das. 2010. Natural language question generation using syntax and keywords. In *Proc. of The Third Workshop on Question Generation (QG)*. questiongeneration.org, 1–10.
Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent Continuous Translation Models.. In *Proc. of EMNLP*, Vol. 3. 413.
Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*. 3294–3302.
K. Knight and D. Marcu. 2000a. Statistics-based summarization - step one: Sentence compression. In *Proc. of AAAI*. 703–710.
Kevin Knight and Daniel Marcu. 2000b. Statistics-based summarization-step one: Sentence compression. *AAAI/IAAI* 2000 (2000), 703–710.
K. Knight and D. Marcu. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence* 139 (2002).
I. Labutov, S. Basu, and L. Vanderwende. 2015. Deep Questions without Deep Understanding. In *Proc. of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. 889–C898.

Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015. A Hierarchical Neural Autoencoder for Paragraphs and Documents. In *Proc. of the 53rd Annual Meeting of the Association for Computational Linguistics*. 1106C1115.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, Vol. 8. Barcelona, Spain.

Hui Lin and Jeff Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 912–920.

D. Lindberg, F. Popowich, J. Nesbit, and P. Winne. 2013. Generating natural language questions to support learning on-line. In *Proc. of the 14th European Workshop on Natural Language Generation*. 105–114.

Konstantin Lopyrev. 2015. Generating news headlines with recurrent neural networks. *arXiv preprint arXiv:1512.01712* (2015).

H.P. Luhn. 1958. The automatic creation of literature abstracts. *I.B.M. Journal of Research and Development* 2 (1958).

M.T. Luong, H. Pham, and C.D. Manning. 2015a. Effective approaches to attention-based neural machine translation. In *Proc. of the Intl. Conf. on Empirical Methods in Natural Language Processing*. 1412–1421.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015b. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

D. Marcu. 1997. From discourse structures to text summaries. In *Proc. of the ACL Workshop on Intelligent Scalable Text Summarization*. 82–C88.

K. Mazidi and R.D. Nielsen. 2014. Linguistic considerations in automatic question generation. In *Proc. of the 52nd Annual Meeting of the Association for Computational Linguistics*. 321–326.

Grainne McKenna, David Hevey, and Elaine Martin. 2010. Patients' and providers' perspectives on bibliotherapy in primary care. *Clinical psychology & psychotherapy* 17, 6 (2010), 497–509.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proc. of the 2004 conference on empirical methods in natural language processing*.

Amit Mishra and Sanjay Kumar Jain. 2016. A survey on question answering systems with classification. *Journal of King Saud University - Computer and Information Sciences* 28, 3 (2016), 345–361.

R. Mitkov and L.A. Ha. 2003. Computer-aided generation of multiple-choice tests. In *Proc. of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing*. 17–22.

J. Mostow and W. Chen. 2009. Generating instruction automatically for the reading strategy of self-questioning. In *Proc. of the 2nd Workshop on Question Generation (AIED)*. 465–472.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Ça glar Gulçehre, and Bing Xiang. 2016. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In *Proc. of CoNLL 2016*. 280.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proc. of the Intl. Workshop on Advances in Neural Information Processing Systems*.

Andrew McGregor abd Arthur C. Graesser Olney and Natalie K. Person. 2012. Question generation from concept maps. *Dialogue & Discourse* 3, 2 (2012), 75C99.

C.D. Paice. 1990. Constructing literature abstracts by computer: Techniques and prospects. *Information Processing and Management* 26 (1990).

C.D. Paice and P.A. Jones. 1993. The identification of important concepts in highly structured technical papers. In *Proc. of ACM SIGIR*. 69–78.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. of the 40th annual meeting on association for computational linguistics*. 311–318.

Wilawan Phornphatcharaphong. 2012. E-Bibliotherapy System: Book Contents for Improving Quality of Youth's Life. *Tem Journal* 1, 3 (2012), 192–199.

Dragomir R Radev, Hongyan Jing, Małgorzata Styś, and Daniel Tam. 2004. Centroid-based summarization of multiple documents. *Information Processing & Management* 40, 6 (2004), 919–938.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proc. of the Intl. Conf. on Empirical Methods in Natural Language Processing*. 2383–2392.

U. Reimer and U. Hahn. 1988. Text condensation as knowledge base abstraction. In *Proc. of the Fourth Intl. Conf. on Artificial Intelligence Applications*. 338–344.

Matthew Richardson, Christopher J.C. Burges, and Erin Renshaw. 2013. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Proc. of Intl. Conf. on Empirical Methods in Natural Language Processing*. 193C203.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *nature* 323, 6088 (1986), 533.

Vasile Rus and Mihai Lintean. 2012. A comparison of greedy and optimal assessment of natural language student input using word-to-word similarity metrics. In *Proc. of the Seventh Workshop on Building Educational Applications Using NLP*. 157–162.

V. Rus, B. Wyse, P. Piwek, M. Lintean, S. Stoyanchev, and C. Moldovan. 2010. The first question generation shared task evaluation challenge. In *Proc. of the 6th Intl. Conf. on Natural Language Generation*. 251–257.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Sentence Summarization. In *Proc. of the 2015 Conference on Empirical Methods in Natural Language Processing*. 379–389.

G. Salton, A. Singhal, M. Mitra, and C. Buckley. 1997. Automatic text structuring and summary. *Information Process and Management* 33 (1997).

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics*. 1715–1725.

I.V. Serban, A. García-Durán, C. Gulcehre, S. Ahn, S. Chandar, A. Courville, and Y. Bengio. 2016. Generating Factoid Questions With Recurrent Neural Networks: The 30M Factoid Question-Answer Corpus. In *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics*. 588C–598.

Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. 2017. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *arXiv preprint arXiv:1706.09799* (2017).

H. Singer and D. Donlan. 1982. Active Comprehension: Problem-Solving Schema with Question Generation for Comprehension of Complex Short Stories. *Reading Research Quarterly* (1982), 166–186.

E.F. Skorokhodko. 1972. Adaptive method of automatic abstracting and indexing. In *Proc. of the IFIP Congress*. 1179–C1182.

Rui Sun, Yue Zhang, Meishan Zhang, and Dong-Hong Ji. 2015. Event-Driven Headline Generation.. In *Proc. of ACL*. 462–472.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075* (2015).

Sho Takase, Jun Suzuki, Naoaki Okazaki, Tsutomu Hirao, and Masaaki Nagata. 2016. Neural Headline Generation on Abstract Meaning Representation. In *Proc. of EMNLP*. 1054–1059.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. From Neural Sentence Summarization to Headline Generation: A Coarse-to-Fine Approach. In *Proc. of the Twenty-Sixth Intl. Joint Conf. on Artificial Intelligence IJCAI*. 4109–4115.

Duyu Tang, Nan Duan, Tao Qin, and Ming Zhou. 2017. Question Answering and Question Generation as Dual Tasks. *arXiv preprint arXiv:1706.02027* (2017).

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2016. Newsqa: A machine comprehension dataset. *CoRR, https://arxiv.org/abs/1611.09830* (2016).

Yuen-Hsien Tseng, Chi-Jen Lin, Hsiu-Han Chen, and Yu-I Lin. 2006. Toward generic title generation for clustered documents. *Information Retrieval Technology* (2006), 145–157.

P.D. Turney. 2000. Learning algorithms for keyphrase extraction. *Information Retrieval* 2 (2000).

Yuya Unno, Takashi Ninomiya, Yusuke Miyao, and Jun'ichi Tsujii. 2006. Trimming CFG parse trees for sentence compression using machine learning approaches. In *Proc. of the COLING/ACL on Main conference poster sessions*. 850–857.

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

Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. 2015. Sequence to sequence-video to text. In *Proc. of the IEEE Intl. Conf. on Computer Vision*. 4534–4542.

Xiaojun Wan and Jianwu Yang. 2008. Multi-document summarization using cluster-based link analysis. In *Proc. of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 299–306.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR* abs/1609.08144 (2016). http://arxiv.org/abs/1609.08144

Yunxing Xin, Yongqiang Chen, Li Jin, Yici Cai, and Ling Feng. 2017. TeenRead: An Adolescents Reading Recommendation System Towards Online Bibliotherapy. *2017 IEEE International Congress on Big Data (BigData Congress)* (2017), 431–434.

Songhua Xu, Shaohui Yang, and Francis Chi-Moon Lau. 2010. Keyword Extraction and Headline Generation Using Novel Word Features.. In *AAAI*. 1461–1466.

Yi Yang, Wen tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *Proc. of Intl. Conf. on Empirical Methods in Natural Language Processing*. 2013C2018.

Xuchen Yao, Gosse Bouma, and Yi Zhang. 2012. Semantics-based question generation and implementation. *Dialogue & Discourse* 3, 2 (2012), 11C42.

Xingdi Yuan, Tong Wang, Caglar Gulcehre, Alessandro Sordoni, Philip Bachman, Sandeep Subramanian, Saizheng Zhang, and Adam Trischler. 2017a. Machine Comprehension by Text-to-Text Neural Question Generation. In *Proc. of the 2nd Workshop on Representation Learning for NLP*. 15–25.

Xingdi Yuan, Tong Wang, Caglar Gulcehre, Alessandro Sordoni, Philip Bachman, Sandeep Subramanian, Saizheng Zhang, and Adam Trischler. 2017b. Machine Comprehension by Text-to-Text Neural Question Generation. *arXiv preprint arXiv:1705.02012* (2017).

D. Zajic, B. Dorr, and R. Schwartz. 2002. Automatic Headline Generation for Newspaper Stories. In *Proc. of the ACL Workshop on DUC*. 78–85.

David Zajic, Bonnie Dorr, and Richard Schwartz. 2004. Bbn/umd at DUC-2004: Topiary. In *Proc. of the HLT-NAACL 2004 Document Understanding Workshop, Boston*. 112–119.

Jianmin Zhang, Tianming Wang, and Xiaojun Wan. 2016. PKUSUMSUM: A Java Platform for Multilingual Document Summarization. In *Proc. of COLING (Demos)*. 287–291.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2018. Neural Question Generation from Text: A Preliminary Study. *Natural Language Processing and Chinese Computing* (January 2018), 662–671.