Automatic Generation of Headlines for Online Math Questions

Ke Yuan, 1,2 Dafang He, 2 Zhuoren Jiang, 3 Liangcai Gao, 1* Zhi Tang, 1* C. Lee Giles 2

1Wangxuan Institute of Computer Technology, Peking University, Beijing, 100080, China
2The Pennsylvania State University, University Park, PA 16802, USA
3School of Data and Computer Science, Sun Yat-sen University, Guangzhou, 510006, China
{yuanke, glc, tangzhi}@pku.edu.cn, duh188@psu.edu, jiangzhhr3@mail.sysu.edu.cn, giles@ist.psu.edu

Abstract

Mathematical equations are an important part of dissemination and communication of scientific information. Students, however, often feel challenged in reading and understanding math content and equations. With the development of the Web, students are posting their math questions online. Nevertheless, constructing a concise math headline that gives a good description of the posted detailed math question is nontrivial. In this study, we explore a novel summarization task denoted as geNERating A concise Math hEadline from a detailed math question (NAME). Compared to conventional summarization tasks, this task has two extra and essential constraints: 1) Detailed math questions consist of text and math equations which require a unified framework to jointly model textual and mathematical information; 2) Unlike text, math equations contain semantic and structural features, and both of them should be captured together. To address these issues, we propose MathSum, a novel summarization model which utilizes a pointer mechanism combined with a multi-head attention mechanism for mathematical representation augmentation. The pointer mechanism can either copy textual tokens or math tokens from source questions in order to generate math headlines. The multi-head attention mechanism is designed to enrich the representation of math equations by modeling and integrating both its semantic and structural features. For evaluation, we collect and make available two sets of real-world detailed math questions along with human-annotated summaries. For evaluation, we collect and make available two sets of real-world detailed math questions along with human-annotated summaries. Experimental results demonstrate that our model (MathSum) significantly outperforms state-of-the-art models for both the EXEQ-300k and OFEQ-10k datasets.

Introduction

Math equations are widely used in the fields of Science, Technology, Engineering, and Mathematics (STEM). However, it is often daunting for students to understand math content and equations when they are reading STEM publications (Liu and Qin 2014; Jiang et al. 2018). Because of the Web, students post detailed math questions online for help. Recent question systems, such as Mathematics Stack

Exchange1 and MathOverflow2, attempt to address this need. From the viewpoint of questioners, the contents of detailed math questions are usually complex and long. In order to efficiently help those who pose the question, it would be helpful to have a headline which is concise and to the point. Correspondingly, those who will answer the question (answerers) also need a clear and brief headline to quickly determine if they should bother to respond. Therefore, giving a concise math headline to a detailed question is important and meaningful. Figure 1 illustrates an example of the question along with its headline posted in Mathematics Stack Exchange3. It’s clear that, a complicated question can make it difficult for answerers to understand the intent of the question, while a concise headline can effectively reduce the cost of this operation.

To this end, we explore a novel approach for geNERating A Math hEadline for detailed questions (NAME). Here, we define the NAME task as a summarization task. Compared to conventional summarization tasks, the NAME task

---

1https://math.stackexchange.com
2https://mathoverflow.net
3https://math.stackexchange.com/questions/3331385
has two extra essential issues that need to be addressed: 1) Jointly modeling text and math equations in a unified framework. Textual (words) and mathematical (equations) information are usually coexisting in detailed questions and brief headlines, as shown in Figure 1. As such, it is natural and necessary to process in some way text and math equations together (Schubotz et al. 2016; Yasunaga and Lafferty 2019). However, it is not evident how to model this in a unified framework. For instance, Yasunaga and Lafferty (Yasunaga and Lafferty 2019) attempted to utilize both text and mathematical representations, but both were treated as separate components. We argue that this approach loses much crucial information, e.g., the position and the semantic dependency between text and equations. 2) Capturing semantic and structural features of math equations synchronously. Unlike text, math equations not only contain semantic features, but also structural features. For instance, equation “\( f = \frac{a}{b} \)” and “\( f b = a \)” have the same semantic features, but different structural features. However, most existing research separately considers only one of these two characteristics. For instance, this work (Yuan et al. 2016; Zanibbi et al. 2016) only considered the structural information of equations for mathematical information retrieval tasks while other work (Deng et al. 2017; Yasunaga and Lafferty 2019) treated a math equation as basic symbols and modeled them as text, which led to structural features loss.

To address these issues, we propose MathSum, a novel method that combines pointers with multi-head attention for mathematical representation augmentation. The pointer mechanism can either copy textual tokens or math tokens from source questions in order to generate math headlines. The multi-head attention mechanism is designed to enrich the representation of each math equation separately by modeling and integrating both semantic and structural features. For evaluation, we construct two large datasets (EXEQ-300k and OFEQ-10k) which contain 290,479 and 12,548 detailed questions with corresponding math headlines from Mathematics Stack Exchange and MathOverflow, respectively. We compare our model with several abstractive and extractive baselines. Experimental results demonstrate that our model significantly outperforms several strong baselines on the NAME task.

In summary, the contributions of our work are:

- an innovative NAME task for generating a concise math headline in response to giving a detailed math question.
- a novel summarization model MathSum that addresses the essential issues of the NAME task, in which the textual and mathematical information can be jointly modeled in a unified framework; while both semantic and structural features of math equations can be synchronously captured.
- novel math datasets\(^4\). To the best of our knowledge, these are the first mathematical content/question datasets associated with headline information.

\(^4\)https://github.com/yuankepku/MathSum

Related Work

Mathematical Equation Representation

Unlike text, math equations are often highly structured. They not only contain semantic features, but also structural features. Recent work (Roy, Upadhyay, and Roth 2016; Zanibbi et al. 2016; Yuan et al. 2018; Jiang et al. 2018) focused mainly on the structural features of math equations, and utilized tree structures to represent equations for mathematical information retrieval and mathematical word problem solving. Other work (Gao et al. 2017; Krstovski and Blei 2018; Yasunaga and Lafferty 2019) instead focused mainly on the semantic features of equations. They processed an equation as a sequence of symbols in order to learn its representation.

Mathematical Equation Generation

Similar to text generation, math equation generation has been widely explored. Recent work (Deng et al. 2017; Zhang, Bai, and Zhu 2019; Le, Indurkhya, and Nakagawa 2019) utilized an end-to-end framework to generate equations from mathematical images, e.g., handwritten math equations. Other work (Roy, Upadhyay, and Roth 2016; Wang et al. 2018) inferred math equations for word problem solving. However, this work only supported limited types of operators (i.e., \(+\), \(-\), \(*\), \(/\)). The work (Yasunaga and Lafferty 2019) most related to ours created a model to generate equations given specific topics (e.g., electric field). Our task (NAME), instead, aims at generating math headlines from both equations and text without clear topics. Thus, our NAME is quite challenging since it requires models to generate correct equations in the correct positions in the generated headlines.

Summarization and Headline Generation

Summarization, a fundamental task in Natural Language Processing (NLP), can be categorized basically into extractive methods and abstractive methods. Extractive methods (Mihalcea and Tarau 2004; Nishikawa et al. 2014) extract sentences from the original document to form the summary. Abstractive methods (See, Liu, and Manning 2017; Tan, Wan, and Xiao 2017a; Narayan, Cohen, and Lapata 2018; Gavrilov, Kalaidin, and Malykh 2019) aim at generating the summary based on understanding the document.

We view headline generation as a special type of summarization, with the constraint that only a short sequence of words is generated and that it preserves the essential meaning of a math question document. Recently, headline generation methods with end-to-end frameworks (Tan, Wan, and Xiao 2017b; Narayan, Cohen, and Lapata 2018; Zhang et al. 2018; Gavrilov, Kalaidin, and Malykh 2019) achieved significant success. Math headline generation is similar to existing headline generation tasks, but still differs in several aspects. The major difference is that a math headline consists of text and math equations which require jointly modeling and inferring text and math equations.
Datasets | avg. math num ques. | head. | avg. text tokens ques. | head. | avg. math tokens ques. | head. | avg. sent. num ques. | head. | text vocab. size ques. | head. | math vocab. size ques. | head.
---|---|---|---|---|---|---|---|---|---|---|---|---
EXEQ-300k | 6.08 | 1.72 | 60.65 | 7.72 | 12.27 | 9.91 | 4.68 | 1.52 | 84,272 | 21,568 | 1,049 | 663
OFEQ-10k | 8.56 | 1.41 | 105.92 | 8.61 | 10.04 | 6.84 | 6.53 | 1.40 | 25,733 | 6,721 | 581 | 393

Table 1: Statistics of the EXEQ-300k and OFEQ-10k (where avg. math num = average math equation number; avg. text tokens = average textual token number; avg. math tokens = average math equation token number; avg. sent. num = average sentence number; text vocab. size = text vocabulary size; math vocab. size = math vocabulary size; ques. = detailed question (source); headl. = math headline (target)).

Task and Dataset

Task Definition
Let us define the NAME task as a summarization one. Let $\mathcal{S} = (s_0, s_1, ..., s_N)$ denote the sequence of the input detailed question. $N$ is the number of tokens in the source, $s \in \{s^w, s^e\}$, $s^w$ represents the textual token (word), and $s^e$ indicates the math token. For each input $\mathcal{S}$, there is a corresponding output math headline with $M$ tokens $\mathcal{Y} = (y_0, y_1, ..., y_M)$ where $y \in \{y^w, y^e\}$ and $y^w, y^e$ are textual tokens and math tokens, respectively. The goal of NAME is to generate a math headline learned from the input question, namely, $\mathcal{S} \rightarrow \mathcal{Y}$.

Dataset
Since this NAME task is new, we could find no public benchmark dataset. As such, we build two real-world math datasets, EXEQ-300k (from Mathematics Stack Exchange) and OFEQ-10k (from MathOverflow), for model training and evaluation. Both datasets consist of detailed questions with corresponding math headlines.

In EXEQ-300k and OFEQ-10k, each question is written in detailed math, and the corresponding headline is a human-written question summary with math equations, typically by the questioner. In Mathematics Stack Exchange and MathOverflow, math equations are enclosed by the “$$” symbols. We use in our datasets “<m>” and “</m>” to replace “$$” in order to indicate the begin and end of an equation. In addition, The toolkit Stanford CoreNLP and \LaTeX{} tokenizer in im2markup are used to tokenize separately the text and equations in questions and headlines.

Specifically, we collect 346,202 pairs of <detailed questions, math headline> from Mathematics Stack Exchange and 13,408 pairs from MathOverflow. To help with analysis and ensure quality, we remove pairs which contain math equations that cannot be tokenized by \LaTeX{} tokenizer. This results in 290,479 pairs from Mathematics Stack Exchange which form EXEQ-300k and 12,548 pairs from MathOverflow which form OFEQ-10k. See Table 1 and Table 2 for more details. In EXEQ-300k, on average there are respectively 6.08 and 1.72 math equations in the question and headlines. In contrast, OFEQ-10k contains more math equations in the question (8.56) and less in the headline (1.41). In EXEQ-300k the questions have 60.65 textual tokens and 12.27 math tokens on average, while the headline has 7.72 textual tokens and 9.91 math tokens on average. Correspondingly, in OFEX-10k, there are on average 105.92 textual tokens and 10.04 math tokens in the question, and on average 10.04 textual tokens and 6.84 math tokens in the headline. Compared to EXEQ-300k, OFEX-10k contains more tokens (textual token and math token) in questions, and less in headlines. From Figure 2, we also see that OFEQ-10k has a higher proportion of novel n-grams than EXEQ-300k. Based on the above observations, we believe that the constructed datasets are significantly different and mutually complementary.

Approach
Here we describe our proposed deep model, MathSum, which we designed for the NAME task.

5Math token is the fundamental element which can form a math equation(Deng et al. 2017)

6https://stanfordnlp.github.io/CoreNLP/
7https://github.com/harvardnlp/im2markup
MathSum Model

As shown in Figure 3, MathSum utilizes a pointer mechanism with a multi-head attention mechanism for mathematical representation augmentation. It consists of two main components: (1) an encoder which jointly learns the representation of math equations and text, and (2) a decoder which learns to generate headlines from the learned representation.

For the encoder, the crucial issue is to build effective representations for tokens in an input question. As mentioned in NAME task, there are two different token types (i.e., textual and math) and their characteristics are intrinsically different. Math tokens not only contain the semantic features (mathematical meaning) but also the structural features (e.g., super/subscript, numerator/denominator, recursive structure). Therefore, the representation learning should vary according to the token type. In this study, we introduce a multi-head attention mechanism to enrich the representation of math tokens.

The token $s_i$ of the input question $S$ is first converted into a continuous vector representation $s_i \in \mathbb{R}^d$, so that the vector representation of the input is $S = [s_0, ..., s_N]$ where $N$ is the number of tokens in the input and $s^w$, $s^m$ are vector representation of textual and math tokens, respectively. Then the vectors of math tokens within an equation are fed into a block with multi-head attention (Vaswani et al. 2017) which then enriches its representation by considering both its semantic and structural features. Please note that each equation in the input will be separately fed into the block since an equation is a fundamental unit for characterizing the semantic and structural features of a series of math tokens. Let $M_k = \{s_i^e, ..., s_j^e\}$ denote the initial vector representation of the $k$-th math equation with $m$ math tokens as input. Then the multi-head attention block transforms the $s_i^e$ to its enriched representation $s_i^e \prime$. This is calculated by

$$s_i^e \prime = f_{\text{Multi-head}}(s_i^e, \{s_j^e, ..., s_j^e+m\}), i \in \{j, ..., j+m\}$$ (1)

where $f_{\text{Multi-head}}$ is the multi-head attention block, $j$ is the beginning index of math equation $M_k$, and $j + m$ is the end index.

After that, the enriched vector representation of the input is $S^e = [s_i^e \prime, ..., s_N^e]$ where $s_i \in \{s^w, s^m\}$ is fed into the update layer (a single-layer bidirectional LSTM) one-by-one. The hidden state $h_i$ is updated according to the previous hidden state $h_{i-1}$ and current token vector $s_i^e \prime$.

$$h_i = f(h_{i-1}, s_i^e \prime)$$ (2)

where $f$ is the dynamic function of LSTM unit and $h_i$ is the hidden state of token $s_i$ in the step $i$.

In the decoder, we aggregate the encoder hidden states $h_0, ..., h_N$ using a weighted sum that then becomes the context vector $C_t$:

$$C_t = \sum_i \alpha_it_i$$ (3)

where

$$\alpha_i = \text{softmax}(e_i)$$

$$e_i = v^T \text{tanh}(W_h h_i + W_h^i h_i^\prime + b_{\text{attn}})$$ (4)

$v, W_h, W_h^i$ and $b_{\text{attn}}$ are the learnable parameters. $h_i^\prime$ is the hidden state of the decoder at time step $t$. The attention $\alpha$ is the distribution over the input position.

$\text{softmax}(\cdot)$ is the softmax function, $\text{tanh}(\cdot)$ is the hyperbolic tangent function.
At this point, the generated math headline may contain textual tokens or math tokens from the source which could be out-of-vocabulary. Thus, we utilize a pointer network (See, Liu, and Manning 2017) to directly copy tokens from source. Considering that the token \( w \) maybe copied from the source or generated from the vocabulary, we use the copy probability \( p_c \) as a soft switch to choose copied tokens from the input or generated textual tokens from the vocabulary.

\[
p(y_t = w | S, y_{<t}) = p_c \sum_{1 \leq w_i \leq V} \alpha_{iw} + (1 - p_c) f(h_t, C_t)
\]

\[
p_c = f(C_t, h_t, x_t)
\]

where \( f \) is non-linear function and \( x_t \) is the decoder input at timestep \( t \).

Finally, the training loss at time step \( t \) is defined as the negative log likelihood of the target word \( w_t^* \) where

\[
Loss_t = -\log p(y_t = w_t^* | S, y_{<t})
\]

Experimental Setup

Comparison of Methods

We compare our model with baseline methods on both the EXEQ-300k and OFEQ-10k for the NAME task. Four extractive methods are implemented as baselines: Random, simply selects a sentence from the input question. Lead, simply selects the leading sentence from the input question, while Tail selects the last sentence, and TextRank extracts sentences from the text according to their scores computed by an algorithm similar to PageRank. In addition, three abstractive methods are also used to compare against Math-Sum. Seq2Seq is a sequence to sequence model based on the LSTM unit and attention mechanism (Bahdanau, Cho, and Bengio 2015). PtGen is a pointer network which allows copying tokens from the source (See, Liu, and Manning 2017). Transformer is a neural network model that is designed based on a multi-head attention mechanism (Vaswani et al. 2017).

Experiment Settings

We randomly split EXEQ-300k into training (90\%, 261,341), validation (5\%, 14,564), and testing (5\%, 14,574) sets. In order to get enough testing samples, we split OFEQ-10k in a 80\% training (10,301), 10\% validation (1,124), and 10\% testing (1,123) proportions.\(^{10}\)

For our experiments, the dimensionality of the word embedding is 128 and the number of hidden states for LSTM units for both encoder and decoder is 512. The multi-head attention block contains 4 heads and 256-dimensional hidden states for the feed-forward part. The model is trained using AdaGrad (Duchi, Hazan, and Singer 2011) with a learning rate of 0.2, an initial accumulator value of 0.1, and a batch size of 16. Also, we set the dropout rate as 0.3. The vocabulary size of the question and headline are both 50,000. In addition, the encoder and decoder share the token representations. At test time, we decode the math headline using beam search with beam size of 3. We set the minimum length as 20 tokens on EXEQ-300k and 15 tokens on OFEQ-10k. We implement our model in PyTorch and train on a single Titan X GPU.

Experimental Results

Quantity Performance

Metrics Here we use three standard metrics: ROUGE (Lin 2004), BLEU (Papineni et al. 2002) and METEOR (Denkowski and Lavie 2014) for evaluation. The ROUGE metric measures the summary quality by counting the overlapping units (e.g., n-gram) between the generated summary and reference summaries. We report the F1 scores for R1 (ROUGE-1), R2 (ROUGE-2), and RL (ROUGE-L). The BLEU score is a widely used as an accuracy measure for machine translation and computes the n-gram precision for R1 (ROUGE-1), R2 (ROUGE-2), and RL (ROUGE-L). The METEOR metric measures the summary quality by counting the overlapping units (e.g., n-gram) between the generated summary and reference summaries. We report the F1 scores for R1 (ROUGE-1), R2 (ROUGE-2), and RL (ROUGE-L). The BLEU and METEOR scores are calculated by using nlg-eval\(^{11}\) package, and ROUGE scores are based on rouge-baselines\(^{12}\) package.

We use the edit distance and exact match to check the similarity of the generated equations compared with the gold standard equations in the math headlines. These two metrics are widely used for the evaluation of equation generation (Deng et al. 2017; Wu et al. 2018). Edit distance quantifies how dissimilar two strings are by counting the minimum number of operations required to transform one string into the other. Based on \( N \) samples in the test set, we use two types of edit distance. One is Edit Distance(m) which is math-level dissimilar score and is defined as \(EditDistance(m) = \sum_{i=0}^{N} \frac{\minMd}{\max([P_i], [G_i])} \), where \( \minMd \) is the minimum edit distance between equations in the generated headline and the gold standard headline, \( |P_i| \) and \( |G_i| \) are the number of equations in the \( i \)-th generated headline and gold headline. The other Edit Distance(s) is the sentence-level dissimilar score, and is formulated as \(EditDistance(s) = \frac{\sum_{i=0}^{N} \minMd}{N} \). Exact Match checks the exact match accuracy between the gold standard math tokens and generated math tokens and is calculated as \(ExactMatch = \frac{\sum_{i=0}^{N} \frac{PM_i \cap GM_i}{\min(\text{NL} \cdot PM_i, GM_i)}}{N} \), where \( PM_i \) and \( GM_i \) are the sets of math tokens in the \( i \)-th generated headline and gold standard headline.

\(^{11}\)https://github.com/Maluuba/nlg-eval
\(^{12}\)https://github.com/sebastianGehrmann/rouge-baselines
<table>
<thead>
<tr>
<th>Models</th>
<th>EXEQ-300k</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>OFEQ-10k</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R1</td>
<td>R2</td>
<td>RL</td>
<td>BLEU-4</td>
<td>METEOR</td>
<td>R1</td>
<td>R2</td>
<td>RL</td>
<td>BLEU-4</td>
</tr>
<tr>
<td>Tail</td>
<td>22.55</td>
<td>14.69</td>
<td>20.76</td>
<td>22.23</td>
<td>23.78</td>
<td>15.46</td>
<td>7.03</td>
<td>13.36</td>
<td>11.13</td>
</tr>
<tr>
<td>Lead</td>
<td>42.23</td>
<td>31.30</td>
<td>39.29</td>
<td>29.89</td>
<td>31.61</td>
<td>27.68</td>
<td>14.92</td>
<td>24.07</td>
<td>14.56</td>
</tr>
<tr>
<td>Seq2Seq</td>
<td>52.14</td>
<td>38.33</td>
<td>49.00</td>
<td>42.20</td>
<td>30.65</td>
<td>38.64</td>
<td>23.42</td>
<td>35.24</td>
<td>27.67</td>
</tr>
<tr>
<td>PtGen</td>
<td>53.26</td>
<td>39.92</td>
<td>50.09</td>
<td>44.10</td>
<td>31.76</td>
<td>40.27</td>
<td>25.30</td>
<td>36.51</td>
<td>28.07</td>
</tr>
<tr>
<td>Transformer</td>
<td>54.49</td>
<td>40.57</td>
<td>50.90</td>
<td>45.79</td>
<td>32.92</td>
<td>40.54</td>
<td>24.36</td>
<td>36.39</td>
<td>28.82</td>
</tr>
<tr>
<td>MathSum</td>
<td>57.53</td>
<td>45.62</td>
<td>54.81</td>
<td>52.00</td>
<td>37.47</td>
<td>42.44</td>
<td>28.15</td>
<td>39.99</td>
<td>29.44</td>
</tr>
</tbody>
</table>

Table 3: Comparison of different models on the EXEQ-300k and OFEQ-10 test sets for F1 scores of R1 (ROUGE-1), R2 (ROUGE-2), RL (ROUGE-L), BLEU-4, and METEOR.

<table>
<thead>
<tr>
<th>Models</th>
<th>Edit Distance(m)</th>
<th>Edit Distance(s)</th>
<th>Exact Match</th>
<th>Edit Distance(m)</th>
<th>Edit Distance(s)</th>
<th>Exact Match</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>8.76</td>
<td>21.84</td>
<td>9.29</td>
<td>7.20</td>
<td>17.73</td>
<td>5.60</td>
</tr>
<tr>
<td>Tail</td>
<td>9.42</td>
<td>20.89</td>
<td>6.65</td>
<td>7.30</td>
<td>14.45</td>
<td>3.47</td>
</tr>
<tr>
<td>Lead</td>
<td>7.47</td>
<td>20.27</td>
<td>12.39</td>
<td>6.58</td>
<td>17.75</td>
<td>6.70</td>
</tr>
<tr>
<td>TextRank</td>
<td>7.68</td>
<td>21.36</td>
<td>12.68</td>
<td>6.75</td>
<td>20.27</td>
<td>7.71</td>
</tr>
<tr>
<td>Seq2Seq</td>
<td>6.68</td>
<td>13.57</td>
<td>13.26</td>
<td>8.69</td>
<td>16.78</td>
<td>8.68</td>
</tr>
<tr>
<td>PtGen</td>
<td>6.59</td>
<td>13.43</td>
<td>13.60</td>
<td>8.06</td>
<td>15.56</td>
<td>8.56</td>
</tr>
<tr>
<td>Transformer</td>
<td>6.32</td>
<td>13.23</td>
<td>13.94</td>
<td>5.56</td>
<td>10.51</td>
<td>8.41</td>
</tr>
<tr>
<td>MathSum</td>
<td><strong>5.82</strong></td>
<td><strong>12.07</strong></td>
<td><strong>15.21</strong></td>
<td>5.71</td>
<td>10.76</td>
<td><strong>8.98</strong></td>
</tr>
</tbody>
</table>

Table 4: Comparison of different models on the EXEQ-300k and OFEQ-10k test sets according to math evaluation metrics. Edit Distance(m) and Edit Distance(s) evaluate those that are dissimilar (the smaller the better). Exact Match is the number of math tokens accurately generated in math headlines (the larger the better).

**Results**  Comparisons of models can be found in Table 3. All models perform better on EXEQ-300k than OFEQ-10k. A possible explanation is that the EXEQ-300k contains a lower proportion of novel n-grams in its gold standard math headlines (illustrated in Figure 2). For extractive models, we find that Lead obtains a good performance on EXEQ-300k, while TextRank performs well on OFEX-10k. Since OFEX-10k contains more sentences for each question, TextRank is more likely to pick out the accurate sentence. Unsurprisingly, abstractive models perform better than extractive models on both datasets. Compared to ordinary Seq2Seq, PtGen gets better performance, since it uses a copying strategy to directly copy tokens from the source question. The transformer can outperform PtGen, which implies that by utilizing multi-head attention mechanism, we obtain a better learning of representation. MathSum significantly outperforms other models for all evaluation metrics on both datasets. Thus, MathSum initially addresses some of the challenges of NAME task and generates satisfactory headlines for questions.

In addition, we also evaluate the gap between the generated headlines and human-written headlines. The Edit Distance(m), Edit Distance(s) and Exact Match scores for different models using EXEQ-300k and OFEQ-10k are shown in Table 4. The results show that extractive models perform worse, if we use the metric Edit Distance(s) instead of Edit Distance(m) for evaluation. Since extractive models directly select sentences from source questions, some selected sentences may not contain math equations. For abstractive baselines, the Transformer obtains the best performance. This observation reinforces the claim that a multi-head attention mechanism can construct a better representation for math equations. On EXEQ-300k, our model, MathSum, achieves the best performance on all metrics. On OFEX-10k, MathSum gets the best performance for Exact Match and second best performance (slightly weaker than Transformer) for Edit Distance(m) and Edit Distance(s). A possible reason is that in OFEX-10k, the lengths of math equations in source questions are usually long, while the ones in headlines are often short. Compared to the Transformer, the copying mechanism could cause MathSum to copy long equations from the source questions, which may result in a slight decreased performance for Edit Distance(m) and Edit Distance(s) metrics.

**Quality Analysis**

**Jointly modeling quality**  The heatmap in Figure 4 visualizes the attention weights from MathSum. Figure 4(a) compares the source detailed question with MathSum. Figure 4(b) compares the source detailed question with the human-written math headline. The generated math headline from Math-
Detailed Question:
In \( \mathbb{CP}^2 \) we define coordinate triangle to be the one with sides \( \{x_0 = 0\}, \{x_1 = 0\} \) and \( \{x_2 = 0\} \). How would you define its interior? What kind of equation should it satisfy?

Human Written Math Headline:
interior of a triangle in \( \mathbb{CP}^2 \)

MathSum Generated Math Headline:
interior of coordinate triangle in \( \mathbb{CP}^2 \).

(a) An example of detailed question

(b) Attention weights for partial source detailed question tokens

Figure 4: Heatmap of attention weights for source detailed questions. MathSum learns to align key textual tokens and math tokens with the corresponding tokens in the source question.

Case study
To gain an insightful understanding regarding the generation quality of our method, we present three typical examples in Table 5. The first two are selected from EXEQ-300k\(^{13,14}\) and the last one is selected from OFEQ-10k\(^{15}\). From the examples, we see that the generated headlines and the human-written headlines have comparability and similarity. Generally, the generated headlines are coherent, grammatical, and informative. We also observe that, it is important to locate the main equations for NAME task. If the generation method emphasizes a subordinate equation, it will generate an unsatisfactory headline, such as the second example in Table 5.

Conclusions and Future Work
Here we define and explore the novel NAME task of automatic headline generation for online math questions using a new deep model, MathSum. Two new datasets (EXEQ-300k and OFEQ-10k) are constructed for algorithm training and testing and are made available. Our experimental results demonstrate that our model can often generate useful math headlines and significantly outperform a series of state-of-the-art models. Future work could focus on enriched representations of math equations for mathematical information retrieval and other math-related research.

Table 5: Examples of generated math headlines given detailed questions.

<table>
<thead>
<tr>
<th>Detailed Question (EXEQ-300k)</th>
<th>Partial Math Detailed Question (EXEQ-300k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partial Math Detailed Question (EXEQ-300k)</td>
<td>So I am asked to find the inverse elements of this set ( \mathbb{Z}[i] = {a + ib</td>
</tr>
<tr>
<td>Human-Written</td>
<td>finding the inverse elements of ( \mathbb{Z}[i] = {a + ib</td>
</tr>
<tr>
<td>MathSum</td>
<td>finding the inverse elements of ( \mathbb{Z}[i] = {a + ib</td>
</tr>
<tr>
<td>Partial Math Detailed Question (EXEQ-300k)</td>
<td>Suppose that the function ( \psi : \mathbb{R}^2 \rightarrow \mathbb{R} ) is continuously differentiable. Define the function ( g : \mathbb{R}^2 \rightarrow \mathbb{R} ) by...</td>
</tr>
<tr>
<td>Human-Written</td>
<td>using the chain rule in ( \mathbb{R}^n )</td>
</tr>
<tr>
<td>MathSum</td>
<td>find ( \frac{\partial g}{\partial s} (s, t) )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Detailed Question (OFEQ-10k)</th>
<th>Partial Math Detailed Question (OFEQ-10k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partial Math Detailed Question (OFEQ-10k)</td>
<td>In the paper of Herbert Clemens Curves on generic hypersurfaces the author shows that for a generic hypersurface ( V ) of ( \mathbb{P}^n ) of sufficiently high degree there is no rational...</td>
</tr>
<tr>
<td>Human-Written</td>
<td>rational curves in ( \mathbb{P}^n ) and immersion</td>
</tr>
<tr>
<td>MathSum</td>
<td>rational curves in ( \mathbb{P}^n )</td>
</tr>
</tbody>
</table>

Acknowledgments
This work is partially supported by China Scholarship Council and projects of National Natural Science Foundation of China (No. 61876003 and 61573028), Guangdong Basic and...
Applied Basic Research Foundation (2019A1515010837) and Fundamental Research Funds for the Central Universities (18lgpy62), and the National Science Foundation.

References


Narayan, S.; Cohen, S. B.; and Lapata, M. 2018. Don’t give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. ACL.


