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]
[1]Wangxuan Institute of Computer Technology, Peking University, Beijing, 100080, China
[2]The Pennsylvania State University, University Park, PA 16802, USA
[3]School of Data and Computer Science, Sun Yat-sen University, Guangzhou, 510006, China
{yuanke, glc, tangzhi}@pku.edu.cn, duh188@psu.edu, jiangzhr3@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 ge**N**erating **A** concise **M**ath h**E**adline 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-written math headlines, namely EXEQ-300k and OFEQ-10k. 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 Exchange*[1] and *MathOverflow*[2], 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 Exchange*[3]. It's clear that, a complicated question can make it difficult for answerers to understand the intent of the questioner, while a concise headline can effectively reduce the cost of this operation.

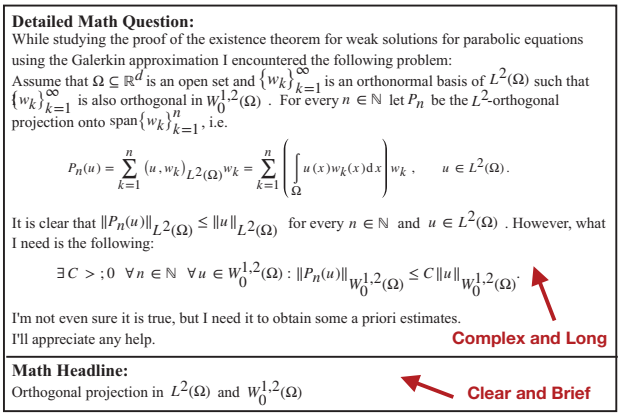


Figure 1: Example of a detailed math question along with its headline. The question is complex and long and the headline is clear and brief.

To this end, we explore a novel approach for ge**N**erating **A** **M**ath h**E**adline for detailed questions (**NAME**). Here, we define the NAME task as a summarization task. Compared to conventional summarization tasks, the NAME task

*are the corresponding authors

[1]https://math.stackexchange.com
[2]https://mathoverflow.net
[3]https://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 "$fb = a$" 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 summarizaton, 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		avg. text tokens		avg. math tokens		avg. sent. num		text vocab. size		math vocab. size	
	ques.	headl.	ques.	headl.	ques.	headl.	ques.	headl.	ques.	headl.	headl.	ques.
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)).

datasets	question pairs	correct question pairs
EXEQ-300k	346,202	290,4794
OFEQ-10k	13,408	12,548

Table 2: Statistics of two datasets (EXEQ-300k and OFEQ-10k) with respect to overall number of collected question pairs and the number of correct question pairs.

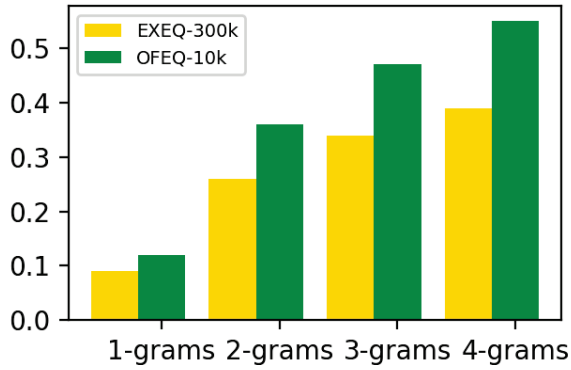


Figure 2: Proportion of novel n-grams for the gold standard math headlines in EXEQ-300k and OFEQ-10k.

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[5]. 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`[6] and LaTeX tokenizer in `im2mark`[7] 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.

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

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

[7]https://github.com/harvardnlp/im2markup

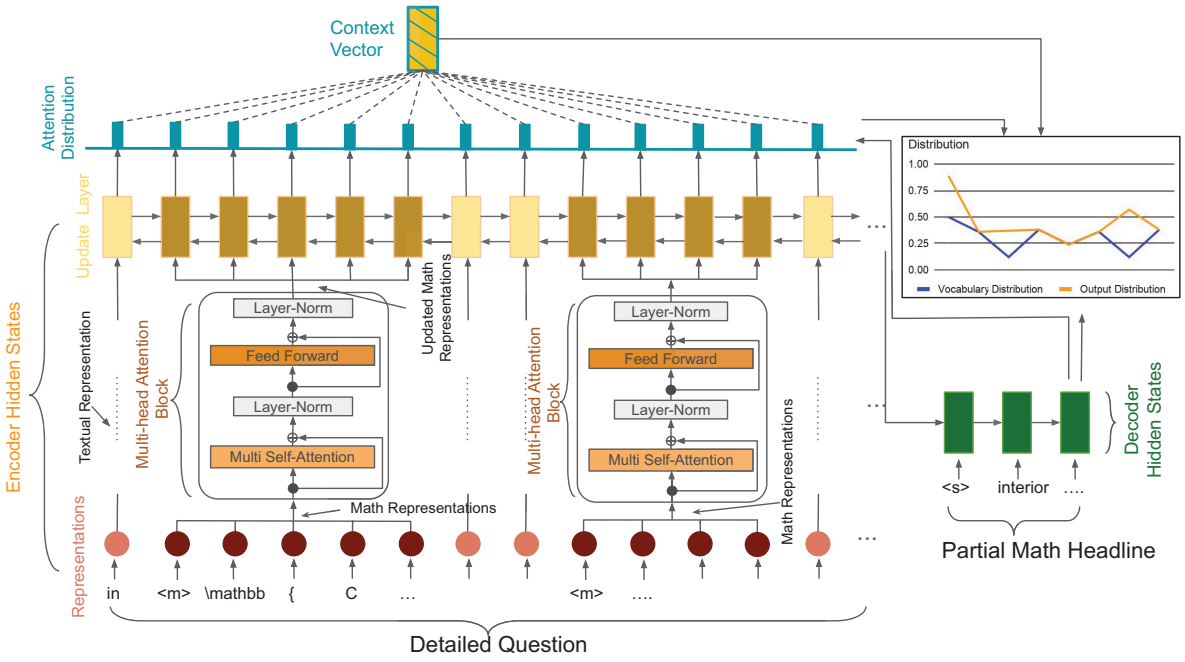


Figure 3: Architecture of MathSum. For a question, each math equation vector representation $[\mathbf{s}_j^e, ..., \mathbf{s}_{j+m}^e]$ will pass through a multi-head attention block to produce a new vector representation $\overline{\mathbf{s}}_j^e$ to $\overline{\mathbf{s}}_{j+m}^e$ which updates the original representation. The updated vector representation $[\mathbf{s}_0^{'}, ..., \mathbf{s}_N^{'}]$ is then fed into an update layer one-by-one.

MathSum Model

As shown in Figure 3, MathSum utilizes a pointer mechanism with a mutli-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, (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/sub script, 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 $\mathcal{S}$ is first converted into a continuous vector representation $\mathbf{s}_i$, so that the vector representation of the input is $\mathbf{S} = [\mathbf{s}_0, ..., \mathbf{s}_N]$ where N is the number of tokens in the input and $\mathbf{s}^w$, $\mathbf{s}^e$ 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 $M_k = \{\mathbf{s}_j^e, ..., \mathbf{s}_{j+m}^e\}$ 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 $\mathbf{s}_i^e$ to its enriched representation $\overline{\mathbf{s}}_i^e$. This is calculated by

$$\overline{\mathbf{s}}_i^e = f_{\text{Multi-head}}(\mathbf{s}_i^e, [\mathbf{s}_j^e, ..., \mathbf{s}_{j+m}^e]), i \in \{j, .., j+m\} \quad (1)$$

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

After that, the enriched vector representation of the input is $\mathbf{S}^{'} = [\mathbf{s}_0^{'}, ..., \mathbf{s}_N^{'}]$ where $\mathbf{s}^{'} \in \{\mathbf{s}^w, \overline{\mathbf{s}}^e\}$ 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 $\mathbf{s}_i^{'}$,

$$h_i = f(h_{i-1}, \mathbf{s}_i^{'}) \quad (2)$$

where f is the dynamic function of LSTM unit and h_i is the hidden state of token $\mathbf{s}^{'}$ 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:

$$C_t = \sum_i \alpha_{it} h_i \quad (3)$$

where

$$\begin{aligned} \alpha_t &= \text{softmax}(\mathrm{e_t}) \\ \mathrm{e_{it}} &= v^{\mathrm{T}} \tanh(\mathrm{W_h h_i} + \mathrm{W_{h'} h_t^{'}} + \mathrm{b_{attn}}) \end{aligned} \quad (4)$$

v,W_h,$W_{h'}$ and b_{attn} are the learnable parameters. $h_t^{'}$ is the hidden state of the decoder at time step t. The attention α is the distribution over the input position.

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.

$$p(y_t = w|\mathcal{S}, y_{<t}) = p_c \sum_{i:w_i=w} \alpha_{it} + (1 - p_c) f(h_t^{'}, C_t)$$
$$p_c = f(C_t, h_t^{'}, x_t) \tag{5}$$

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 likehood of the target word w_t^* where

$$Loss_t = -\log p(y_t = w_t^*|\mathcal{S}, y_{<t}) \tag{6}$$

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**, randomly 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**[8] extracts sentences from the text according to their scores computed by an algorithm similar to PageRank. In addition, three abstractive methods[9] are also used to compare against MathSum. **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 of a candidate sequence to the reference. METEOR is recall-oriented and evaluates translation hypotheses by aligning them to reference translations and calculating sentence-level similarity scores. 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 $EditDistance(m) = \sum_{i=0}^{N} \frac{minMd_i}{\max(|P_i|,|G_i|)}$, 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 $EditDistance(s) = \frac{\sum_{i=0}^{N} minMd_i}{N}$. Exact Match checks the exact match accuracy between the gold standard math tokens and generated math tokens and is calculated as $ExactMatch = \frac{\sum_{i=0}^{N}(PM_i \& GM_i)}{N}$, where PM_i and GM_i are the sets of math tokens in the i-th generated headline and gold standard headline.

[8]For TextRank, we use the implementation in summanlp, https://github.com/summanlp/textrank

[9]We use the implementation of OpenNMT, https://github.com/OpenNMT/OpenNMT-py

[10]For a fair comparison, all models used the same experimental data setup. For EXEQ-300k, all models are trained and tested on the same dataset. For OFEQ-10k, in order to achieve better experimental results, all models are first trained on the training set of EXEQ-300k, then fine-tuned and tested using OFEQ-10k

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

[12]https://github.com/sebastianGehrmann/rouge-baselines

Models	EXEQ-300k					OFEQ-10k				
	R1	R2	RL	BLEU-4	METEOR	R1	R2	RL	BLEU-4	METEOR
Random	31.56	21.35	28.99	24.32	23.40	22.95	11.48	19.85	13.19	18.00
Tail	22.55	14.69	20.76	22.23	23.78	15.46	7.03	13.36	11.13	11.68
Lead	42.23	31.30	39.29	29.89	31.61	27.68	14.92	24.07	14.56	20.99
TextRank	42.19	30.85	38.99	28.29	31.78	29.66	16.41	25.59	14.20	23.71
Seq2Seq	52.14	38.33	49.00	42.20	30.65	38.64	23.42	35.24	27.67	25.27
PtGen	53.26	39.92	50.09	44.10	31.76	40.27	25.30	36.51	28.07	25.90
Transformer	54.49	40.57	50.90	45.79	32.92	40.54	24.36	36.39	28.82	25.89
MathSum	**57.53**	**45.62**	**54.81**	**52.00**	**37.47**	**42.44**	**28.15**	**38.99**	**29.44**	**26.84**

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.

Models	EXEQ-300k			OFEQ-10k		
	Edit Distance(m)	Edit Distance(s)	Exact Match	Edit Distance(m)	Edit Distance(s)	Exact Match
Random	8.76	21.84	9.29	7.20	17.73	5.60
Tail	9.42	20.89	6.65	7.30	14.45	3.47
Lead	7.47	20.27	12.39	6.58	17.75	6.70
TextRank	7.68	21.36	12.68	6.75	20.27	7.71
Seq2Seq	6.68	13.57	13.26	8.69	16.78	8.68
PtGen	6.59	13.43	13.60	8.06	15.56	8.56
Transformer	6.32	13.23	13.94	**5.56**	**10.51**	8.41
MathSum	**5.82**	**12.07**	**15.21**	5.71	10.76	**8.98**

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 mutli-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

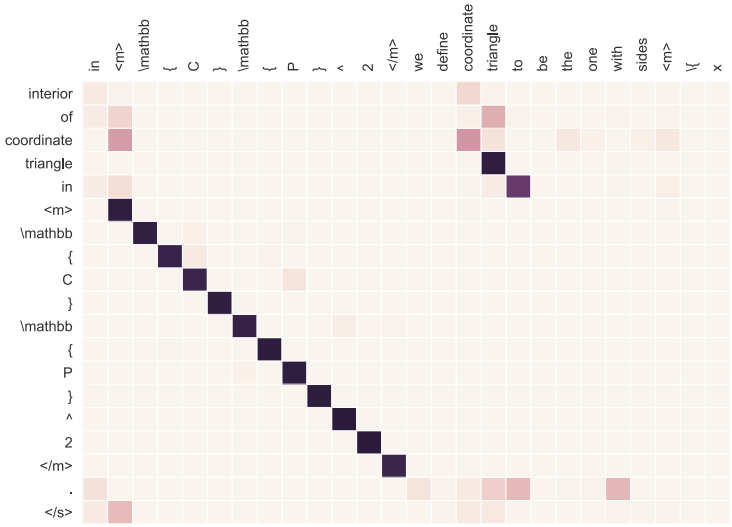
Jointly modeling quality The heatmap in Figure 4 visualizes the attention weights from MathSum. Figure 4(a) compares the source detailed question with its human-written math headline and the generated math headline from Math-

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 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.

Sum. As Figure 4 shows, there are both textual tokens and math tokens in the generated headline. Note that both math tokens and textual tokens can be effectively aligned to their corresponding tokens in the source. For instance, the textual tokens "coordinate", "triangle" and the math tokens "P", "C" are both all successfully aligned.

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.

[13]https://math.stackexchange.com/questions/2431575
[14]https://math.stackexchange.com/questions/752067
[15]https://mathoverflow.net/questions/291434

Examples	
Partial Math Detailed Question (EXEQ-300k)	So I am asked to find the inverse elements of this set $\mathbb{Z}[i] = \{a + ib \vert a, b \in \mathbb{Z}\}$ (I know that this is the set of Gaussian integers). I was pretty much do...
Human-Written	finding the inverse elements of $\mathbb{Z}[i] = \{a + ib \vert a, b \in \mathbb{Z}\}$
MathSum	finding the inverse elements of $\mathbb{Z}[i] = \{a + ib \vert a, b \in \mathbb{Z}\}$
Partial Math Detailed Question (EXEQ-300k)	Suppose that the function $\psi : \mathbb{R}^2 \to \mathbb{R}$ is continuously differentiable. Define the function $g : \mathbb{R}^2 \to \mathbb{R}$ by...
Human-Written	using the chain rule in $\mathbb{R}^n$
MathSum	find $\frac{\partial g}{\partial s}(s, t)$
Partial Math Detailed Question (OFEQ-10k)	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...
Human-Written	rational curves in $\mathbb{P}^n$ and immersion
MathSum	rational curves in $\mathbb{P}^n$

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

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

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. *In proceedings of ICLR*.

Deng, Y.; Kanervisto, A.; Ling, J.; and Rush, A. M. 2017. Image-to-markup generation with coarse-to-fine attention. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 980–989. JMLR. org.

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

Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12(Jul):2121–2159.

Gao, L.; Jiang, Z.; Yin, Y.; Yuan, K.; Yan, Z.; and Tang, Z. 2017. Preliminary exploration of formula embedding for mathematical information retrieval: can mathematical formulae be embedded like a natural language? *arXiv preprint arXiv:1707.05154*.

Gavrilov, D.; Kalaidin, P.; and Malykh, V. 2019. Self-attentive model for headline generation. In *European Conference on Information Retrieval*, 87–93. Springer.

Jiang, Z.; Gao, L.; Yuan, K.; Gao, Z.; Tang, Z.; and Liu, X. 2018. Mathematics content understanding for cyberlearning via formula evolution map. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 37–46. ACM.

Krstovski, K., and Blei, D. M. 2018. Equation embeddings. *arXiv preprint arXiv:1803.09123*.

Le, A. D.; Indurkhya, B.; and Nakagawa, M. 2019. Pattern generation strategies for improving recognition of handwritten mathematical expressions. *arXiv preprint arXiv:1901.06763*.

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

Liu, X., and Qin, J. 2014. An interactive metadata model for structural, descriptive, and referential representation of scholarly output. *Journal of the Association for Information Science and Technology* 65(5):964–983.

Mihalcea, R., and Tarau, P. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, 404–411.

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*.

Nishikawa, H.; Arita, K.; Tanaka, K.; Hirao, T.; Makino, T.; and Matsuo, Y. 2014. Learning to generate coherent summary with discriminative hidden semi-markov model. In *Proceedings of COLING 2014*, 1648–1659.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, 311–318.

Roy, S.; Upadhyay, S.; and Roth, D. 2016. Equation parsing: Mapping sentences to grounded equations. *EMNLP*.

Schubotz, M.; Grigorev, A.; Leich, M.; Cohl, H. S.; Meuschke, N.; Gipp, B.; Youssef, A. S.; and Markl, V. 2016. Semantification of identifiers in mathematics for better math information retrieval. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 135–144. ACM.

See, A.; Liu, P. J.; and Manning, C. D. 2017. Get to the point: Summarization with pointer-generator networks. *ACL*.

Tan, J.; Wan, X.; and Xiao, J. 2017a. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 1171–1181.

Tan, J.; Wan, X.; and Xiao, J. 2017b. From neural sentence summarization to headline generation: A coarse-to-fine approach. In *IJCAI*, 4109–4115.

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

Wang, L.; Zhang, D.; Gao, L.; Song, J.; Guo, L.; and Shen, H. T. 2018. Mathdqn: Solving arithmetic word problems via deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Wu, J.-W.; Yin, F.; Zhang, Y.-M.; Zhang, X.-Y.; and Liu, C.-L. 2018. Image-to-markup generation via paired adversarial learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 18–34. Springer.

Yasunaga, M., and Lafferty, J. 2019. Topiceq: A joint topic and mathematical equation model for scientific texts. *AAAI*.

Yuan, K.; Gao, L.; Wang, Y.; Yi, X.; and Tang, Z. 2016. A mathematical information retrieval system based on rankboost. In *Proceedings of the 16th ACM/IEEE-CS on Joint Conference on Digital Libraries*, 259–260. ACM.

Yuan, K.; Gao, L.; Jiang, z.; and Tang, Z. 2018. Formula ranking within an article. In *Proceedings of the 18th ACM/IEEE-CS on Joint Conference on Digital Libraries*, 123–126. ACM.

Zanibbi, R.; Davila, K.; Kane, A.; and Tompa, F. W. 2016. Multi-stage math formula search: Using appearance-based similarity metrics at scale. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 145–154. ACM.

Zhang, W.; Bai, Z.; and Zhu, Y. 2019. An improved approach based on cnn-rnns for mathematical expression recognition. In *Proceedings of the 2019 4th International Conference on Multimedia Systems and Signal Processing*, 57–61. ACM.

Zhang, R.; Guo, J.; Fan, Y.; Lan, Y.; Xu, J.; Cao, H.; and Cheng, X. 2018. Question headline generation for news articles. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 617–626. ACM.