生成，推理与排序：基于多任务架构的数学文字题生成

曹天旸[1,2*]，**许晓丹** [1,2*]，**常宝宝** [1†]
[1] 北京大学计算语言学教育部重点实验室，北京 100871
[2] 北京大学软件与微电子学院，北京 102600
{ctymy, diane1968, chbb}@pku.edu.cn

摘要

数学文字题是一段能反映数学等式潜在逻辑的叙述性文本。成功的数学问题生成在语言生成和教育领域都具有广阔的应用前景。前人的工作大多需要人工标注的模板或关键词作为输入，且未考虑数学表达式本身的特点。本文提出了一种多任务联合训练的问题文本生成模型。我们设计了三个辅助任务，包括数字间关系抽取、数值排序和片段替换预测。他们与生成目标联合训练，用以监督解码器的学习，增强模型对运算逻辑和问题条件的感知能力。实验证明所提方法能有效提升生成的数学文字题的质量。

关键词： 数学文字题生成；多任务学习

Generating, Reasoning & Ranking: Multitask Learning Framework for Math Word Problem Generation

Tianyang Cao[1,2*], **Xiaodan Xu**[1,2*], **Baobao Chang**[1†]
[1]Key Laboratory of Computational Linguistics, Ministry of Education, Peking University, Beijing 100871, China
[2] School of Software and Microelectronics, Peking University, Beijing 102600, China
{ctymy, diane1968, chbb}@pku.edu.cn

Abstract

A math word problem (MWP) is a narrative which reflects the underlying logic of math equations. Successful MWP generation has wide prospect in language generation and educational field. Previous works mostly require human-annotated templates or topic words, besides, they fail to consider the characteristics of MWP. This paper proposes a multitask learning based MWP generation framework. We devise three novel tasks, including number relation extraction, number ranking and sentence substitution prediction. These tasks are jointly trained with generation objective and supervise the learning of MWP decoder while enhancing the model' s comprehension of arithmetic logic and condition. Experiments demonstrate the effectiveness of our proposed method in equation consistency of generated MWPs.

Keywords: Math word problem generation , Multitask Learning

1 引言

[*] 共同一作
[†] 通讯作者

第二十一届中国计算语言学大会论文集，第178页-第189页，南昌，中国，2022年10月14日至16日。
(c) 2022 中国中文信息学会计算语言学专业委员会

自然语言生成是自然语言处理中的一个重要领域，它的目的是生成流畅、可读性强且忠实于源输入的自然语言文本。在本论文中，我们主要聚焦于一种相对较新的数据文本生成任务—从等式生成数学文字题。公式到数学文本生成的目标是根据给定的等式，自动生成连贯且能够反映其运算逻辑的叙述性文本。如表 1所示是两个范例，其中包括了输入的数学表达式及其对应的数学问题文本。数学问题的生成涉及到内容规划等生成领域基础性课题，因而具有技术发展层面的意义；同时，数学问题的自动生成能够实现中小学数学问题的自动命题，有利于降低教师的教学负担，在计算机辅助教学领域有着广泛的应用前景。同时从学术的角度，该任务的定义还有进一步的拓展空间，例如如何处理符号更复杂的大学阶段数学问题，如何生成更加个性化、多样化的问题表达等等。

传统的数据-文本生成任务，往往是以作为结构化数据的表格记录或一系列三元组作为输入，因而输入的各部分之间是按照语言顺序或者时间顺序组织，而公式到问题文本的生成，其输入是由常量、未知变量、运算符三种符号组成的、不具有明显语义的抽象表达式，并且存在数学运算的逻辑。因此它与传统的生成任务有较大区别，模型也需要特殊的设计。针对该任务，前人的研究还存在很大不足。现有的一些工作 (Zhou and Huang, 2019; Wang et al., 2021; Liu et al., 2020) 在利用神经生成模型实现数学文字题自动生成方面取得了一定的效果。这些工作大多基于表达式（或表达式模板）和若干个话题词来生成数学文字题，他们更关注如何将关键词内容融入到解码过程中去，并反映数学问题发生的场景，而忽视对等式的结构特性及数字、变量间运算关系的理解。

数学表达式	$equ : (1 - 1/3 - 9/20) * x = 245$
数学文字题	At a local high school, ***1/3*** of the students are freshmen, ***9/20*** are juniors. And ***245*** are seniors. ***Find the total number of students.***
数学表达式	$equ : 45/(x - y) = 5$ $equ : 45/(x + y) = 3$
数学文字题	A boat travels ***45*** mi upstream (against the current) in ***5*** h . The boat travels the same distance downstream in ***3*** h. ***What is the rate of the boat in still water.***

Table 1: 数学文字题生成任务示例

首先，在数学文字题中，数字的角色非常重要，数字或者未知变量（x, y, z 等）通常代表现实场景中的某种物理量，例如物品的数量、种类、测度，交通工具的速度等属性。从直观上讲，数学文字题中的任意两个数字间都可能存在一定的逻辑关系，例如“…农场里饲养了 8 只动物，其中 3 只鸡，5 只兔子…”这段表述中，3 和 5 的关系是并列（分别表示鸡和兔子的数量），因而在叙述问题时，应该体现出这两种动物数目的求和；而在表 1的第一个例子中，x 和 1/3 的关系是相乘，因此在叙述问题时，应体现 x 是总体而 1/3 是比例系数。而现有的模型难以捕捉问题文本中数字所对应的物理量间的运算关系，因而对于结构较复杂，包含数字较多的公式，生成效果比较差。其次，在以往的工作中，数学文字题中的数字常常被替换成特殊符号（如 NUM0,NUM1）等，没有考虑数值隐含信息。而实际上数学文字题中，数字间的大小关系对于引导生成也有积极意义，具体有两方面：(1) 数字的大小能够反映一部分的语义信息，例如数字 a 比数字 b 更大，往往会出现数字 a 和 b 的差，或者从一堆数量为 a 的物品中拿走数量为 b 的物品这样的表达。(2) 数字间的大小关系蕴含着一些生活常识，例如电影院/公园所售门票中，成人票售价通常比儿童票更贵；船在航行时其顺水速度会大于逆水速度（顺水速度为船速度 + 水速度，逆水速度为船速度-水速度）；一个数值非常大的数字一般不太可能表达人的年龄，或者一支铅笔的价格等。最后，数学文字题一般由对背景的描述性句子、描述条件的语句和设问句组成，每个句子在文字题中都有其特定的功能。由于缺乏对句子结构的组织和规划，基线模型生成的语句常常逻辑比较混乱，如重复之前时刻的条件，产生和问题语境不相关的词汇或提问对象错误等。

针对以上问题，受前人在生成任务中引入多任务学习的启发 (Ge et al., 2021; Shen et al., 2021), 我们提出了多任务架构的数学表达式-文字题生成模型，将数学问题生成与数学问题理解融合到一个统一的框架中。我们基于 Transformer 构建我们的生成模型，并设计了数字关系抽取、数字排序和片段替换预测三个辅助任务。数字关系抽取中两个数字间的关系定义为他们在数学表达式树上的最近公共祖先（Least Common Ancestor），其中数学表达式树是等式所对应

第二十一届中国计算语言学大会论文集，第178页-第189页，南昌，中国，2022年10月14日至16日。

的后缀树，如图 1所示是找到数字间关系的例子，(a)(b) 都是方程组

$$\begin{cases} 2000 * (1 + 0.04)^5 = x \\ x - 2000 = y \end{cases} \tag{1}$$

对应的表达式树（pseudo root 是用于连接多棵树的虚拟节点），(c) 是方程 $51 * x + 8 * y = 510$ 对应的表达式树。绿色代表作为树的叶子的两个数字节点，而红色结点代表它们的最近公共祖先节点。数字关系抽取要求模型根据生成的问题文本预测其中两个数字间的运算关系标签，以此帮助解码器更好地组织物理量之间的运算逻辑，得到符合实际的表述。数字排序采用自回归的方式，对生成的数学文字题中的数字按照从小到大的顺序进行排序，以期使模型理解如何对公式中的数字进行组织，并感知到生成的问题语句是否合理，对更高质量的生成起到激励作用。片段替换预测则以句子为单位，将参考问题文本句中某一个句子按特定规律替换成另一个片段，在训练时提供给解码器，最终利用一个指针网络模块预测出被替换部分的左右边界。

上述三个辅助任务与标准生成任务进行联合训练。在基于 Dolphin_18K 拓展的数据集上的实验证明，所提出的方法在各项指标上均超越了基线模型，且能有效提升问题文本的逻辑描述与给定表达式之间的一致性。

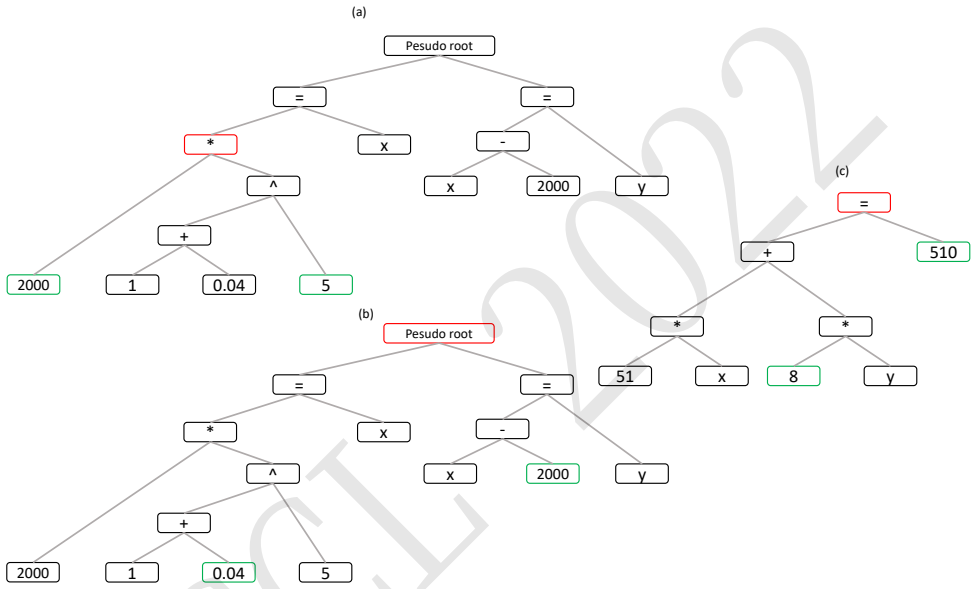

Figure 1: 表达式树上两数字节点的最近公共祖先

2 相关工作

数学文字题生成：早期的公式到文本生成工作主要是基于模板规则的方法，利用问答集编程技术（Answer Set Programming）(Polozov et al., 2015) 及框架语义技术 (Singley and Bennett, 2002; Deane, 2003) 等对模板中的插槽进行填充。利用深度学习架构，(Zhou and Huang, 2019; Wang et al., 2021) 的工作都是基于等式模板和关键词序列进行生成，他们的工作以等式模板和关键词作为输入，其中关键词是使用启发式规则直接从标准的问题文本中进行提取得到。模型使用端到端的方式进行训练，并在解码过程中融合模板和关键词两部分特征。然而这些方法在测试阶段也需要来自正确答案的关键词作为额外输入，这在现实场景中是不可用的。而本文工作采取了更符合实际的设定，即只根据数学等式进行问题文本的生成。(Liu et al., 2020) 的论文将输入的表达式抽象成莱文图，并用外部知识图谱子图引导生成和主题相关的句子。但该方法只能处理线性的表达式，且对于每个等式都需要额外的话题标注。

数据到文本生成：数据到文本生成是将结构化的数据转化为描述性文本 (Siddharthan, 2001; Gatt and Krahmer, 2018)。例如，(Puduppully et al., 2019a; Puduppully et al., 2019b; Gong et al., 2019; Wiseman et al., 2017) 关注于体育比赛新闻报道的生成，(Chisholm et al., 2017;

Equation 1:

$$equ: x + y = 400 \; equ: 2 * x + 3 * y = 1050$$

Problem 1:
[1-11] The attendance at a baseball game was 400 people . [12-22] Student tickets cost $ 2 and adult tickets cost $ 3 . [23-29] Total ticket sales were $ 1050 . [30-38] How many tickets of each type were sold .
Relation Extraction:
r(2,3) = "+" r(400,1050) = "pseudo root"
Number Ranking:
2 < 3 < 400 < 1050
Problem After Replacement:
[1-11] The attendance at a baseball game was 400 people . [12-22] Student tickets cost $ 2 and adult tickets cost $ 3 . [23-38] *The red rose theater sells tickets for $ 4 . 50 and $ 6 . 00* [39-47] How many tickets of each type were sold .
Span Boundary:
[23,38]

Equation 2:

$$equ: (x + 150 * 0.8) / (x + 150) = 0.9$$

Problem 2:
[1-17]If x ounces of pure acid are added to 150 ounces of an 80% acid solution . [18-27]The concentration of the new mixture is 90% acid . [28-45]Find the number of ounces that were added to the original solution to produce the 90% solution .
Relation Extraction:
r(150,0.8) = "*" r(150,0.9) = "="
Number Ranking:
0.8 < 0.9 < 150
Problem After Replacement:
[1-15] *Results of a survey of fifty students indicate that 30 like red jelly beans .* [16-25] The concentration of the new mixture is 90% acid . [26-43] Find the number of ounces that were added to the original solution to produce the 90% solution .
Span Boundary:
[1,15]

Figure 2: 数据集中的两个文字题及其对应的三个辅助任务预测目标

Lebret et al., 2016) 等人的工作面向人物简历的生成，(Zhao et al., 2018; Gao et al., 2020) 等考虑结构化信息，从资源描述符三元组集合生成文本。此外，前人的工作在模型中设计了内容选择和文本规划机制以决定哪些内容应该被表述及按照怎样的顺序表述 (Puduppully et al., 2019a; Perez-Beltrachini and Lapata, 2018)。

多任务文本生成：近年来，多任务学习在文本生成和语言预训练领域得到了广泛应用。(Ge et al., 2021) 等人的工作基于上下文语句和被引用论文的摘要生成论文中的引用句。他们将作者写作引用句的原因分为 4 类，并联合训练生成器和分类判别器对引用的功能进行识别。(Shen et al., 2021) 在表达式解析任务中，联合训练了生成模型和排序打分模型以提升模型对错误表达式树的区分能力。

3 任务定义

我们的系统以一个或多个数学表达式 $\{E_1, E_2, ..., E_{|E|}\}$ 为输入，每个等式都由一系列的数学符号构成:$E_k = x_1 x_2 ... x_{|E_k|}$，其中 $|E_k|$ 是第 k 个公式的长度，由符号的数目来衡量。每个数学符号由以下三种符号构成：数学运算符，例如 "+，-，*，=，）" 等，数字，例如 "0.2，1,30" 等，变量，例如 "x, y, z" 等。任务的输出是一个数学问题文本：$\mathbf{y} = y_1 y_2 ... y_L$，该问题可以用输入的等式解决。$L$ 是问题文本的长度。我们的模型目标是根据输入的公式和之前时刻生成的词 $\boldsymbol{y}_{<t}$ 估计接下来生成的词的条件概率:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \prod_{t=1}^{L} P(y_t|\boldsymbol{y}_{<t}, E_1, E_2, ...) \tag{2}$$

4 模型结构

我们提出的多任务训练的问题文本生成模型如图 3所示。它包括一个标准的 Transformer 生成模型、一个用于预测生成的文字题中数字两两之间关系的**数字关系抽取**模块、一个对生成的文字题中的数字进行排序的**数字排序**模块以及一个对问题文本中被替换片段边界进行定位的**替换预测模块**。生成任务与所有辅助任务共享相同的编码器和解码器，且目标函数进行联合训练。辅助任务只在训练阶段有效。

4.1 以 Transformer 为基础的编码器-解码器模型

序列到序列生成模型是目前生成任务的主流框架。输入的公式序列用 $E = (x_1, x_2, ..., x_n)$ 表示, 编码器由若干个 Transformer 层组成, 使用双向的自注意力机制将 E 映射到一系列连续的向量表示 $R = (\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_n)$:

$$(\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_n) = Transformer_{ENC}(x_1, x_2, ..., x_n) \tag{3}$$

解码器同样也包括多个 Transformer 层, 其中增加了以编码器的输出为关注值的多头交叉注意力模块。解码器每次吸收一个词 s_i，利用前面时刻解码器的输出状态和编码器的输出表示 $\mathbf{R}$ 预测下一时刻词的分布:

$$P(*) = softmax(\mathbf{d}_i\mathbf{W} + \mathbf{b}) \tag{4}$$

$$\mathbf{d}_i = Transformer_{Dec}(\mathbf{R}, s_0, s_1, ..., s_{i-1}) \tag{5}$$

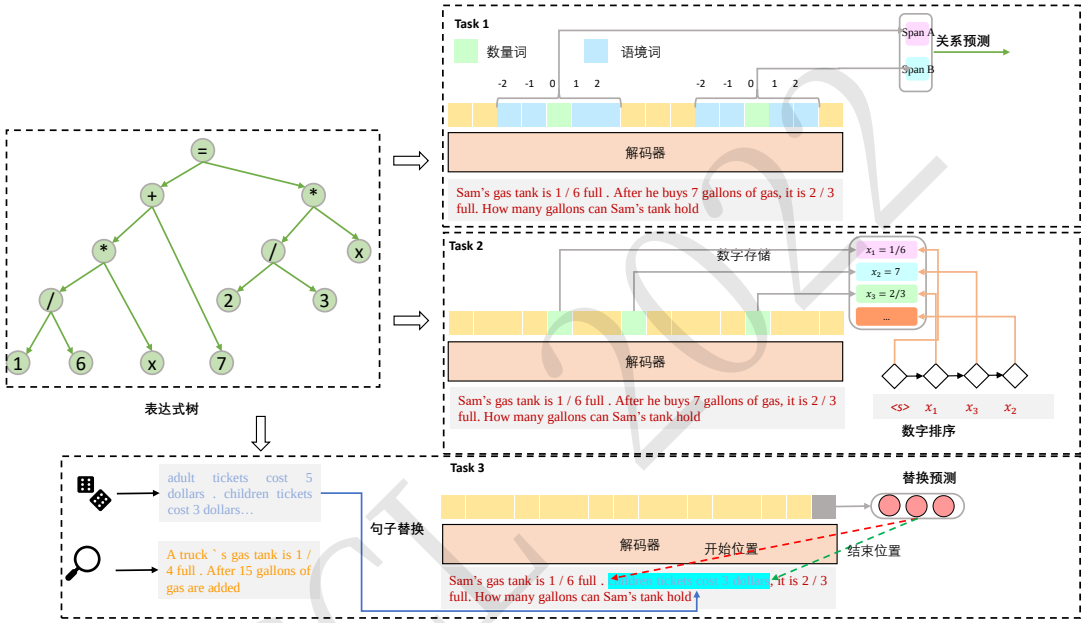

Figure 3: 基于多任务训练的数学问题生成模型

4.2 多任务训练

任务 # 1: 数字关系抽取: 数字关系抽取是利用生成的文字题中两数字的语境表示，对他们间的运算关系进行预测，从而帮助解码器在生成过程中感知到物理量之间的交互。数字间关系的标签类别包括 $\{+, -, *, /, \wedge\ , =, \surd, pseudo\ root\}$ 共 8 种，这些都是可能成为表达式后缀树上两叶子结点最近公共祖先的运算符或特殊标记。对于公式中的两个数字 x_a 和 x_b，他们在数学文字题中出现的位置分别为 p_a 和 p_b，我们将解码器视作对于生成的问题文本的编码器，并使用解码器输出状态序列中数字周围语境词的聚合信息作为该数字的表示。具体来说，对于 x_a，我们首先获得以 p_a 为中心，长度为 3,5,7 的片段（位置范围分别为 $p_a - 1 \sim p_a + 1, p_a - 2 \sim p_a + 2, p_a - 3 \sim p_q + 3$）的表示:

$$\mathbf{c}_{ak} = \text{MLP}([\mathbf{d}_{p_a-k}; \mathbf{d}_{p_a+k}; \mathbf{d}_{p_a-k} \odot \mathbf{d}_{p_a+k}])\ k \in \{1, 2, 3\} \tag{6}$$

其中 $\odot$ 代表逐元素乘积。随后我们学习一个参数 $\mathbf{u} \in \mathbb{R}^d$，通过分别比较 $\mathbf{u}$ 和 $\mathbf{c}_{a1}, \mathbf{c}_{a2}, \mathbf{c}_{a3}$ 来获得长度分别为 3，5，7 的片段的重要程度打分, 并利用得分作为注意力系数融合三个片段的表

示，得到数字 x_a 的最终表示：

$$att_i = \frac{\sigma(\mathbf{u}^T \mathbf{c}_{ai})}{\sum_{j\in\{1,2,3\}} \sigma(\mathbf{u}^T \mathbf{c}_{aj})} \tag{7}$$

$$\mathbf{c}_a^* = \sum_{i\in\{1,2,3\}} att_i \mathbf{c}_{ai} \tag{8}$$

其中 $\sigma(\cdot)$ 代表 sigmoid 函数。用同样的方式也可以得到 x_b 的表示 $\mathbf{c}_b^*$，于是分类的函数和数字关系抽取部分的损失函数可以定义为：

$$P(r(x_a, x_b)|\mathbf{c}_a^*, \mathbf{c}_b^*) \propto \exp(\mathbf{w}_1^T \sigma(\mathbf{W}_2 [\mathbf{c}_a^*; \mathbf{c}_b^*; |\mathbf{c}_a^* - \mathbf{c}_b^*|; \mathbf{c}_a^* \odot \mathbf{c}_b^*])) \tag{9}$$

$$\mathcal{L}_{relation} = \frac{1}{|\Omega|} \sum_{x_a, x_b \in \Omega, x_a \neq x_b} -\log P(r(x_a, x_b)|\mathbf{c}_a^*, \mathbf{c}_b^*) \tag{10}$$

其中 $\mathbf{w}_1$ 和 $\mathbf{W}_2$ 都是参数, 为了简便起见省略了偏置项。$\sigma(\cdot)$ 是一个激活函数，如 ReLU(·) 等。Ω 代表所有同时在数学公式和问题文本中出现的数字的集合，$r(x_a, x_b)$ 代表 x_a 和 x_b 在表达式树上的最近公共祖先。

任务 # 2: 数字排序：如前所述，数学文字题中数字的大小关系能够指示该数字可能代表的对象，同时也蕴含了现实场景中的一些隐含知识, 针对数字数值的比较和排序可以增强模型对于题目条件的理解能力。为此，我们提出数字排序模块用以监督解码器的学习。我们仿照任务 1 中的方法得到数字的特征表示，并将数学文字题中所出现的数字的表示构成一个存储，作为排序模块的输入。排序模块采用递归式生成，按从小到大的顺序，依次预测出最小数字在存储中的位置，第二小的数字在存储中的位置…直到最大的数字在存储中的位置。

具体地，将问题文本中的数字，按照下标从小到大依次记为 $z_1, z_2, ..., z_K$，其中 K 是数字数目，它们的特征表示分别为 $\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_K$。$H = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_K]$ 构成了数字的存储。我们希望生成一个指针序列，按照这些数字升序的顺序依次指向 $[1, K]$ 中的某一个位置。假设把 $z_1, z_2, ..., z_K$ 升序排序后，按下标记作 $z_{p_1} < z_{p_2} < \ < z_{p_K}$，其中 $p_1, p_2, ..., p_K$ 是下标。生成器通过一个两层的 GRU 单元和指针网络实现，我们依次把排序过后数字的向量表示提供给 GRU，以自回归的形式生成下一个更大的数字在 H 中的位置。GRU 单元用零向量初始化，解码的过程可以形式化地写成：

$$\overline{\mathbf{h}}_t = GRU(\overline{\mathbf{h}}_{t-1}, \left[\mathbf{h}_{p_{t-1}}; \mathbf{r}^*\right]) \tag{11}$$

其中在 $t = 0$ 时，$\mathbf{h}_{p_{t-1}}$ 是开始标记 [sos] 的嵌入向量。$\mathbf{r}^*$ 代表对公式表示进行平均池化的全局向量：$\mathbf{r}^* = Meanpool([\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_n])$。随后我们用指针网络预测 z_{p_t} 在 H 中的位置：

$$score(t, i) = \sigma(\mathbf{W}_3 \overline{\mathbf{h}}_t)(\sigma(\mathbf{W}_4 \mathbf{h}_i))^T \ 1 \leq i \leq K \tag{12}$$

$$P(q_t|q_1, q_2, ..., q_{t-1}, H) = softmax_i(score(t, i)) \tag{13}$$

其中 $\sigma(\cdot)$ 是激活函数,q_t 代表从小到大第 t 小的数（也就是 z_{p_t} 在 H 中的正确位置。最后，数字排序任务的损失可以写成：

$$\mathcal{L}_{rank} = -\frac{1}{K} \sum_{t=1}^{K} P(q_t|q_1, q_2, ..., q_{t-1}, H) \tag{14}$$

任务 # 3: 片段替换预测：前人的一些工作将随机选择的答案作为和输入无关的负样本，然后通过打分排序模型对每个输入-输出对赋予一个 $[0, 1]$ 之间的打分, 使模型学习鉴别低质量的回复。在本节中，我们对这种方法进行拓展，一方面，仅仅通过随机选择的句子作为和输入构成错误的公式-问题对可能对于模型效果提高有限，因为随机选择的数学文字题往往和真实的参考答案完全不相关，对于模型来说鉴别比较容易，因此我们希望模型判别更复杂的情形，即生成的句子能够在一定程度反映输入公式的逻辑，但是不完全正确。另一方面，我们希望模型在打分时能够进一步学习到问题文本中哪一段叙述不合理，或不符合场景。

为此，我们设计了一个片段替换的预测任务，它的目标是通过一定的随机策略，将数学文字题中的某一个句子替换成一个完全不相关或者有部分相关性的其他文字题中的句子，再将被替换后的问题文本提供给解码器作为输入，希望模型能够定位被替换的这个句子的范围。具体来说，对于由 N 个句子组成的参考数学问题文本 $P = \{S_0, S_1, ..., S_{N-1}\}$，我们按照如下规则进行替换：

- 在 1/3 的概率下，不对正确的问题文本进行替换。
- 在 1/3 的概率下，随机在训练数据集中选择一个问题文本 $P' = \left\{S'_0, S'_1, ..., S'_{N'-1}\right\}$，然后随机选择 $i \in [0, N-1]$,$j \in [0, N'-1]$，将原始题目中的第 i 个句子替换成 P' 中的第 j 个句子，替换后的数学文字题变为：$P = \left\{S_0, S_1, ..., S_{i-1}, S'_j, S_{i+1}, ..., S_{N-1}\right\}$。
- 在 1/3 的概率下，我们首先通过 BertScore (Zhang* et al., 2020) 工具，在训练数据集中检索一个和参考答案嵌入表示相似度最高的问题文本 $R' = \left\{S''_0, S''_1, ..., S''_{N'-1}\right\}$，然后随机选择 $i \in [0, N-1]$,$j \in [0, N''-1]$，将原始题目中的第 i 个句子替换成 R' 中的第 j 个句子，替换后的数学文字题变为：$P = \left\{S_0, S_1, ..., S_{i-1}, S''_j, S_{i+1}, ..., S_{N-1}\right\}$。

完成替换后，我们将新的数学问题文本 P 作为解码器的监督信号，在 P 的最后附加一个标记 [eos] 作为结束符，这样文本总长度记为 M。解码器输出的状态依次表示为 $\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_M$。我们用最后一个状态 $\mathbf{d}_M$ 和输入公式表示向量 $\mathbf{r}^*$ 拼接作为查询向量，再通过指针网络预测替换部分的范围边界。用 $start^*$ 和 end^* 分别表示被替换部分的开始位置和结束位置的下标。特别地对于上述（1）中原问题文本没有被替换的情形，我们增加一个可学习的向量 $\mathbf{v}$ 拼在 $\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_M$ 前，即 $D = [\mathbf{v}, \mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_M]$，此时 $start^* = end^* = 0$。指针网络预测开始和结束位置的表达式分别为：

$$P(start = i) = softmax(\mathbf{d}_i \mathbf{W}_4 [\mathbf{d}_M; \mathbf{r}^*]) \; 0 \leq i < M \tag{15}$$

$$P(end = j) = softmax(\mathbf{d}_j \mathbf{W}_5 [\mathbf{d}_M; \mathbf{r}^*]) \; i < j < M \tag{16}$$

我们把选择某一片段的概率定义为选中其开始位置和结束位置的概率乘积，这样替换任务的损失函数可以表示为：

$$P(start = i, end = j) = P(start = i) * P(end = j) \tag{17}$$

$$\mathcal{L}_{span} = -\log P(start = start^*, end = end^*) \tag{18}$$

在计算上述 $P(start = i, end = j)$ 时，会对所有可能的片段的得分进行归一化。

模型损失函数模型的损失函数由生成部分的对数似然（用 MLE 表示）和三个辅助学习任务的损失构成：

$$\mathcal{L}_{total} = MLE + \alpha(\mathcal{L}_{relation} + \mathcal{L}_{rank} + \mathcal{L}_{span}) \tag{19}$$

5 模型实验与分析

5.1 数据来源

我们的数据集基于 Dolphin_18K (Huang et al., 2017), 该数据集从 Yahoo!Answer 网站上爬取得到。由于 (Huang et al., 2017) 仅仅开源了 Dolphin_18K 的一个子集（3154 条样本），这对于生成模型的训练来说是不够的。因此我们复用 (Huang et al., 2017) 中给出的脚本从 Yahoo!Answer 上爬取并收集了额外的数据，将数据集的规模扩充到了 14943 个样例（代码和数据集将在论文录用后公开）。对于获得的数据我们进行了预处理，删除了问题文本长度超过 45 个词或低于 15 个词的公式-文本对，最终保留了 9643 个样例。表 2给出了数据集的统计信息。

5.2 基线模型和参数设置

我们将所提出的方法与以下基线模型对比：（1）**Seq2seq** (Bahdanau et al., 2014) 首先被用于机器翻译任务。在本工作中，我们实现了使用注意力机制和拷贝机制的 seq2seq 模型。（2）

	Train	Dev	Test
Size	7714	964	965
Equation Length (average)	16.69	16.23	16.63
Problem Length (average)	28.90	29.64	28.74
Tokens	7445	3065	2875

Table 2: 数据集统计信息

词向量维度	256	Transformer 层数	2
Transformer 隐层维度	256	Transformer 前馈网络中间层维度	512
GRU 隐层维度	256	GRU 层数	2
Adam β_1	0.99	Adam β_2	0.999
Batchsize 大小	32	学习率	e-4
Dropout Rate	0.2	(19) 式中的 α	1

Table 3: 模型超参数设置

SeqGAN (Yu et al., 2016) 基于生成式对抗网络，使用强化学习在每一步生成时评估所得到完整序列的得分期望。在文本生成和音乐生成等多个任务上都取得了提升。(3) **DeepGCN** (Guo et al., 2019) 是深度图卷积神经网络，由于数学公式可以转化成后缀表达式树，这样树上每个节点可以看作图的节点，同时在树上相邻的两节点间进行连边，于是公式文本生成问题可以转化为图到序列的生成的问题。(4) **Transformer** (Vaswani et al., 2017) 被广泛应用于生成任务的模型。(5) **DualCG** (Wei et al., 2019)，在本文中我们使用 DualCG 来把表达式解析和问题文本生成集成到一个统一的框架中。(6) **BART** (Lewis et al., 2020) 是使用标准 transformer 架构的强有力的预训练模型，我们在数学问题生成的数据集上对 BART 进行微调。

模型使用 Adam 优化器进行训练。生成任务和三个辅助任务共享相同的编码器和解码器。第三个任务中在获取训练数据时采取了类似于 Roberta 中动态遮蔽的动态策略，即替换和被替换的内容不是在预处理时决定，而是在执行每一轮次的训练时都运行一次随机替换，这种策略扩大了选取错误句子时的搜索空间，避免解码器反复看到相同的模式，实际上起到了数据扩充的作用。其余参数设置见表 3。

模型	**BLEU**	**ROUGE-L**	**BERTScore**	**METEOR**	**Dist1**(%)	**Dist2**(%)	**NR**(%)
Seq2seq (Bahdanau et al., 2014)	2.59	20.25	82.98	18.51	14.56	34.99	47.60
SeqGAN (Yu et al., 2016)	2.62	19.22	82.56	17.63	12.96	30.02	44.00
DeepGCN (Guo et al., 2019)	3.04	20.94	83.07	19.48	16.81	45.17	49.21
Transformer (Vaswani et al., 2017)	3.14	21.84	83.81	20.26	12.94	43.51	44.84
DualCG (Wei et al., 2019)	3.60	21.43	83.99	20.63	15.47	46.01	40.97
BART_{large} (Lewis et al., 2020)	4.15	22.26	**86.35**	22.30	12.77	46.76	43.47
Full Model	**4.20**	**23.13**	84.61	**22.32**	**19.03**	**53.89**	**71.06**
T1+T2	3.22	20.93	84.90	25.33	10.74	35.62	49.37
T1+T3	4.10	22.52	84.74	22.43	18.90	51.95	70.16
T2+T3	3.92	22.64	84.92	21.96	19.70	53.13	65.64
T1	3.37	21.87	84.57	20.70	16.60	47.63	70.52
T2	3.48	22.39	84.32	21.50	20.90	57.37	69.62
T3	3.71	21.68	84.25	21.37	20.67	57.42	67.81

Table 4: 本文方法和基线模型在主要评测指标上的性能对比。其中 NR 代表数字召回率，Trans 是 Transformer 的缩写

5.3 主要实验结果

自动评测：我们使用了以下自动评测指标：BLEU（BLEU-1 和 BLEU-2 值的平均）(Papineni et al., 2002), ROUGE-L (Lin, 2004), METEOR (Banerjee and Lavie, 2005), BERTScore (Zhang* et al., 2020)（一种基于嵌入表示相似度的文本生成评价指标），Dist-1，Dist-2（指示不同的一元组/二元组在所有一元组/二元组中的比例），数字召回率（衡量正确的问题文本中的数字有多大比例被正确拷贝了）。主要结果如表 4所示，其中也包含只保留部分辅助任务的消融实验。Full Model 表示完整的模型，T1,T2,T3 分别表示三个任务。

可以看到，使用完整的预训练任务设定，我们的模型在 BLEU、ROUGE-L、METEOR 指标上相比标准的 Transformer 模型分别取得了 33.7%、5.9% 和 10.2% 的提升，在 BERTScore 指标方面也取得了 0.8 个百分点的提升，这说明了所设计的学习任务增强了解码器对未来产生的语句的感知和搜索能力。由于针对数字的排序赋予了模型认知数值大小的能力，我们的模型在数字召回方面也表现更好。此外，我们发现完整的模型框架在自动评测指标方面取得了优于

模型	BLEU	ROUGE-L	BERTScore	METEOR	Dist1(%)	Dist2(%)	NR(%)
BART_{large}	4.15	22.26	86.35	22.30	12.77	46.76	43.47
BART_{large}+T1+T2+T3	**4.83**	**23.01**	86.24	**22.60**	**16.92**	**49.68**	**43.37**

Table 5: 在 BART 模型上增加辅助任务所获得的提升

BART 的效果，这说明针对数学公式的特点设计多任务学习目标能有效提升数学问题生成的语言质量。

为了探究多任务联合训练中每个任务所起的作用，我们进行了详细的消融实验，即三个辅助任务中任取两个或只选取一个，并报告了实验结果。如图所示，在只有任务一（关系抽取）和任务二（数值排序）的情况下，模型在 BLEU、ROUGE-L 和 METEOR 方面的表现分别下降 23.3%、5.42% 和 8.64%，在 BERTScore 得分上下降 0.06 个百分点；在只使用任务一（关系抽取）和任务三（片段替换预测）的情况下，模型得分相比完整设置下降不明显；在只使用任务二（数值排序）和任务三（片段替换预测）的情况下，模型在 BLEU、ROUGE-L 和 METEOR 方面的表现分别下降 6.67%、2.64% 和 1.61%。当只采用一种辅助训练任务时，自动评测指标得分均显著低于完整的模型，从而证明了联合训练的优势。

问题类别	测试集中占比%	本文模型	Transformer
1 数字基本运算，人口问题	9.02	6.40	7.19
2 几何类问题	17.72	5.59	4.99
3 概率问题，币值问题	6.84	**3.21**	2.41
4 数字基本运算	14.51	**4.64**	3.03
5 溶液类问题，物质类问题	9.33	**0.79**	0.24
6 金融类问题	8.08	**6.31**	2.47
7 百分比问题，几何类问题	8.08	**6.31**	2.47
8 销售问题	12.64	**4.27**	1.68
9 行程问题	10.36	**3.09**	1.79

Table 6: 在不同话题类型的测试集子集上，多任务架构和标准 Transformer 的 BLEU 值对比

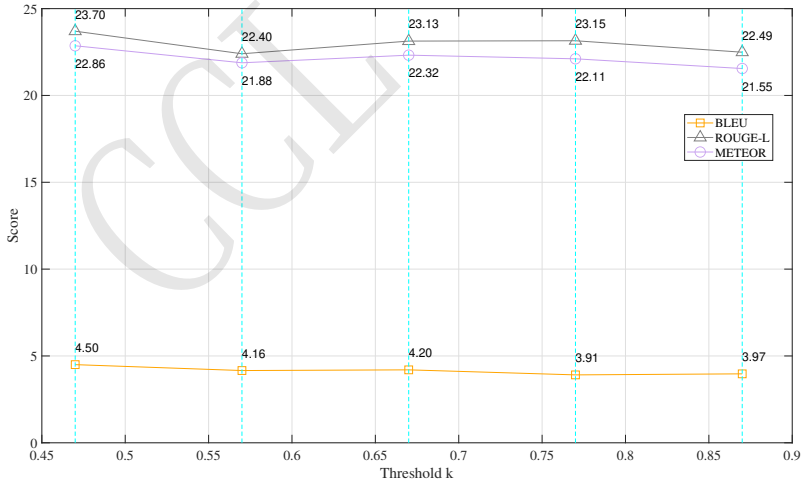

Figure 4: 任务三中 2、3 两种替换策略被采用的概率发生变化时，模型性能的变化

模型泛化能力: 为了验证所提出方法的泛化性能，我们将本文方法应用到 BART 生成模型上，即在微调 BART 的训练过程中加入三个辅助任务的目标函数。实验结果如表 5所示，可以发现在使用了多任务学习之后，BART 的效果获得了进一步提升。这验证了数字关系抽取和数学排序等任务能使生成模型更准确地表述物理量之间的逻辑关系，也说明针对任务特点设计的学习目标有助于增强通用生成模型的泛化能力。

不同类型问题上的实验分析: 为了进一步探究所提出的模型在不同子集上的性能，我们采

第二十一届中国计算语言学大会论文集，第178页-第189页，南昌，中国，2022年10月14日至16日。

用 (Shen et al., 2021) 中对问题文本话题的定义，根据所隶属的话题的不同将测试集划分为 9 个子集。属于不同话题的问题文本在文字表达方面有自身的特点，同时不同子集中高频的数学表达式也具备类别特征，例如存款、利息类问题常常涉及到增长率的幂次计算。相关结果如表 5.3 所示。表 5.3 中的第一列给出了不同问题类别的大致描述（根据该类别所包含的关键词人工归纳，其中数字基本运算是指不涉及具体生活场景，单纯描述几个数之间运算的问题），并给出了测试集中这 9 种类别占比。通过在细分子集上本文方法和标准 Transformer 模型的对比，可以发现：（1）我们的模型在第二个类别（几何类问题）上得分最高，在第六个类别（金融类问题）上得分最低，而基线模型也是如此。这可能是由于几何类问题涉及元素比较单一，遵循相似的模板，因而学习难度较低；而金融类问题包含较多的专用术语，给生成带来了一些困难。（2）本文的方法在 3∼9 类别上相比基线模型 BLEU 值均更高，而在类别 1、2 上 BLEU 值有所下降。尤其是我们的模型在销售、行程类问题子集上提升均比较明显，可能与这类问题需要较多数量关系的理解与推理有关，比如行程与速度的关系、单价与总价的关系、不同类别商品价格的关系等。

模型分析：进一步地，我们对于任务三中采取不同策略替换原问题中的片段对最终实验结果的影响进行探究，以验证所提出模型的鲁棒性。考虑阈值 k 和一个在 [0,1] 区间内符合均匀分布的数，当该数落在 $[0, 1/3]$ 时不进行替换，落在 $[1/3, k]$ 时以训练集中随机抽取问题文本作为不相关句的来源，落在 $[k, 1]$ 时以 BERTScore 检索出的问题文本作为不相关句的来源。默认情况下 k 取 2/3 即 0.67。当 k 分别取 0.47、0.57、0.67、0.77 和 0.87 时，分析模型的 BLEU、ROUGE-L 和 METEOR 指标的变化，并绘成折线图，如图 4所示，其中垂直的蓝色虚线代表了 k 的 5 个采样点。可以看到：（1）当 k 的取值发生变化时，模型的生成质量保持在较高水平，验证了所提出方法的稳定性。（2）总体而言当 k 的取值增加时，评测指标得分有小幅度的降低。这是由于 k 的取值越小，采用第三个策略的概率越大，就有更大的机会选取检索到的问题文本，即用于替换的错误片段和原文强相关，对模型来说造成的干扰较大，也更难区分。这样，在片段预测任务中，模型能够学习识别和真实句子语境类似，但不符合公式逻辑的错误片段，而不仅仅是能够对随机采样的句子进行定位。

	流畅度		**一致性**		**S1**(%)	**S2**(%)
	score	κ	score	κ		
本文模型	**3.97**	0.436	**4.06**	0.497	**32**	**57**
Seq2seq	3.78	0.256	3.48	0.483	23	34
SeqGAN	3.75	0.305	3.28	0.520	20	40
DeepGCN	3.61	0.295	3.55	0.494	29	52
Transformer	3.80	0.333	3.53	0.421	20	45
DualCG	3.88	0.346	3.66	0.455	28	53
BART_{large}	3.56	0.398	3.73	0.454	31	52

Table 7: 生成的数学文字题的人工评测结果

人工评价：为了更好的衡量所提出模型的实际生成质量，我们请了三位人工标注者来判断不同模型给出的结果的质量，其中采用了以下四个方面的评价。(1) 流畅度：流畅度主要衡量生成的数学文字题是否流畅，是否存在语法错误。(2) 一致性：一致性用于衡量数学文字题在文本层面是否连贯 (3) 可解决性 (S1)：由于生成的目标是数学文字题，我们需要考虑该问题是否能被解决，也就是有多大比例的生成的问题文本，可以根据它列出和原数学表达式相同（或者等效）的等式。(4) 可解决性 (S2)：是一个相对于可解决性 (S1) 更宽松的指标。它只要求列出的是一个合法的表达式，而不要求与给定等式相符。

我们随机挑选了 100 个生成的数学问题文本，并且按照五级打分制进行打分。我们把得分映射到 1∼5，而更高的分数代表更好的性能。为了说明不同评分者给出的结果间的一致性，我们使用 Kappa 系数 (κ 值) 来进行评估，κ 值越高说明打分可信度越高。平均后的得分如表 7所示。可以看到所提出的多任务模型的得分无论在流畅性、一致性还是在可解决性方面，得分都是最高的。其中在流畅度、一致性、S1、S2 上比 DualCG 分别提升了 2.32%、10.93%、14.28%、7.55%；比 BART_{large} 分别提升了 11.52%、8.84%、3.23%、9.61%。

6 总结

我们在数学文字题生成的研究中，提出了三个与生成任务联合训练的辅助任务，从物理量之间关系的预测、数值大小的比较以及无关片段的预测三个角度，使模型学习到数学应用题中的常见表述，进而提升了通用生成模型在该任务上的表现。

致谢

本文工作受到国家自然科学基金（61876004、61936012）支持，特此致谢。

参考文献

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv: Computation and Language*.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: an automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization@ACL 2005, Ann Arbor, Michigan, USA, June 29, 2005*, pages 65–72. Association for Computational Linguistics.

Andrew Chisholm, Will Radford, and Ben Hachey. 2017. Learning to generate one-sentence biographies from wikidata. 1:633–642.

Paul Deane. 2003. Automatic item generation via frame semantics: Natural language generation of math word problems. 12.

Hanning Gao, Lingfei Wu, Po Hu, and Fangli Xu. 2020. Rdf-to-text generation with graph-augmented structural neural encoders. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3030–3036. ijcai.org.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *J. Artif. Intell. Res.*, pages 65–170.

Yubin Ge, Ly Dinh, Xiaofeng Liu, Jinsong Su, Ziyao Lu, Ante Wang, and Jana Diesner. 2021. BACO: A background knowledge- and content-based framework for citing sentence generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1466–1478, Online. Association for Computational Linguistics.

Li Gong, Josep Crego, and Jean Senellart. 2019. Enhanced transformer model for data-to-text generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 148–156. Association for Computational Linguistics, November.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019. Densely connected graph convolutional networks for graph-to-sequence learning, 08.

Danqing Huang, Shuming Shi, Chin-Yew Lin, and Jian Yin. 2017. Learning fine-grained expressions to solve math word problems. In *EMNLP*, pages 805–814. Association for Computational Linguistics, September.

Remi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. pages 1203–1213.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880. Association for Computational Linguistics, July.

Chinyew Lin. 2004. Rouge: A package for automatic evaluation of summaries. pages 74–81.

Tianqiao Liu, Qian Fang, Wenbiao Ding, Zhongqin Wu, and Zitao Liu. 2020. Mathematical word problem generation from commonsense knowledge graph and equations. *CoRR*, abs/2010.06196.

Kishore Papineni, Salim Roukos, Todd Ward, and Weijing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. pages 311–318.

Laura Perez-Beltrachini and Mirella Lapata. 2018. Bootstrapping generators from noisy data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1516–1527. Association for Computational Linguistics, June.

Oleksandr Polozov, Eleanor O' Rourke, Adam M. Smith, Luke Zettlemoyer, Sumit Gulwani, and Zoran Popović. 2015. Personalized mathematical word problem generation. In *IJCAI 2015*, May.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019a. Data-to-text generation with content selection and planning. *AAAI 2019*, 33(01):6908–6915.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019b. Data-to-text generation with entity modeling. *ACL 2019*, pages 2023–2035.

Jianhao Shen, Yichun Yin, Lin Li, Lifeng Shang, Xin Jiang, Ming Zhang, and Qun Liu. 2021. Generate & rank: A multi-task framework for math word problems. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2269–2279. Association for Computational Linguistics.

Advaith Siddharthan. 2001. Ehud reiter and robert dale. *Building Natural Language Generation Systems*. cambridge university press, 2000. $64.95/£37.50 (hardback), 234 pages. *Nat. Lang. Eng.*, (3):271–274.

Mark Singley and Randy Bennett. 2002. Item generation and beyond: Applications of schema theory to mathematics assessment. 01.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. pages 5998–6008.

Zichao Wang, Andrew Lan, and Richard Baraniuk. 2021. Math word problem generation with mathematical consistency and problem context constraints. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5986–5999. Association for Computational Linguistics, November.

Bolin Wei, Ge Li, Xin Xia, Zhiyi Fu, and Zhi Jin. 2019. Code generation as a dual task of code summarization. In *NeurIPS*, pages 6559–6569.

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. 2017. Challenges in data-to-document generation. pages 2253–2263.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2016. Seqgan: Sequence generative adversarial nets with policy gradient. *arXiv: Learning*.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *ICLR*. OpenReview.net.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *EMNLP*, pages 3901–3910. Association for Computational Linguistics, October-November.

Qingyu Zhou and Danqing Huang. 2019. Towards generating math word problems from equations and topics. *INLG 2019*, pages 494–503.