Numerical Embeddings for Reasoning over Text and Tables

Article · February 2023

CITATION
1

READS
1,351

2 authors, including:

Ramil Yarullin
HSE University
6 PUBLICATIONS **74** CITATIONS

SEE PROFILE

Numerical Embeddings for Reasoning over Text and Tables

Ramil Yarullin*
HSE University
Yandex

Sergei Isaev*
HSE University

Abstract

The task of answering complex questions over a given context is especially challenging when it involves numerical and discrete reasoning. It requires the ability to work with numbers and capture their meaning. In this work, we first revisit the widely used NumNet model with improvements and an ablation study in which we reveal the redundancy of its reasoning module when using large pre-trained encoders. In light of the limitations of current models, we propose **A**ttention-**e**nhanced **N**umerical **E**mbeddings for **R**easoning (AeNER) and show its effectiveness on the DROP, RACENum, and TAT-QA datasets with textual and tabular contexts. Our experiments indicate that AeNER has strong reasoning capabilities, achieving remarkable 88.88 F1 on DROP and 83.2 F1 on TAT-QA.

1 Introduction

Machine reading comprehension (MRC) is a long-studied task that aims to enable machines to comprehend a given context and answer questions based on it. In a particularly challenging setting of this problem, inferring the answer requires multi-step reasoning with discrete operations such as counting, numerical comparison, sorting, and performing arithmetic calculations. Numerical reasoning is essential for natural language understanding in both technical (Zhu et al., 2021; Chen et al., 2021; Jin et al., 2019) and general (Rajpurkar et al., 2016; Dua et al., 2019) domains.

Consider the example from the TAT-QA dataset (Zhu et al., 2021) in Figure 1. To answer the question "*What is the average net income from 2017-2019?*", a model has to derive the numbers from the corresponding table cells, count the average, and determine the correct scale from the paragraph. In another example from DROP (Dua et al., 2019), one needs to find the relevant numbers and compute the difference between the second largest and the second smallest number.

In DROP and TAT-QA, numeric tokens account for 8.7% and 10.2% of tokens, respectively. Existing numerical MRC models (Zhu et al., 2021; Zhou et al., 2021, 2022a) tend to treat numbers the same as other words. However, multiple studies (Wallace et al., 2019; Naik et al., 2019; Zhang et al., 2020) suggest that the embeddings learned through standard language model pre-training are inefficient for capturing numeracy, especially when extrapolating to larger numbers.

One approach to process numbers is NumNet (Ran et al., 2019), a graph-based neural network that initializes a graph with numbers as nodes and iteratively updates the node representations. The NumNet's enhanced version, NumNet+v2[1], currently being the best-performing model on DROP with open-source code, has been adopted as a strong baseline in multiple MRC benchmark datasets (Ferguson et al., 2020; Zhu et al., 2021; Mishra et al., 2022) with different data formats and domains. In our work, we make two important observations. First, in several studies, the performance of NumNet+v2 is vastly underestimated due to a lack of relevant prediction operations. In particular, Zhu et al. (2021) report that the TagOP model surpasses NumNet+v2 on TAT-QA by 14.4 F1, but our implementation of NumNet shows an 18.7 F1 performance advantage over TagOP. Second, we find that the effect of the NumNet's graph-based reasoning module, while still visible for the BERT-base encoder (Devlin et al., 2019), becomes negligible for large versions of RoBERTa (Liu et al., 2019) and DeBERTa-v3 (He et al., 2021a,b).

In order to better encode the numbers, particularly for use in larger encoders, we propose attention-enhanced numerical embeddings for reasoning (AeNER), a novel method for numerical reasoning over text and tables. We probe various methods for representing numbers (Spithourakis and

* Equal contribution.

[1] https://github.com/llamazing/numnet_plus

Fiscal Year 2019 Cash Flows

	December 27, 2019	December 28, 2018	December 29, 2017
Net income	**$24,193**	**$20,402**	**$14,366**
Non-cash charges	$47,625	$38,186	$28,725
Changes in capital	$26,811	$13,506	$11,594
Cash in activities	$44,154	$33,688	$42,406

... Net cash provided by operations was $45.0 million for fiscal 2019 consisting of $24.2 **million** of net income and $47.6 million of non-cash charges...

Question: What is the **average** net income from 2017-2019?

NumNet+: 19653.67 million
AeNER: 19653.67 thousand
Answer: (**24,193** + **20,402** + **14,366**) / **3** = 19653.67 **thousand**

Passage:
The 2010 United States Census reported that Lassen County had a population of 34,895. The racial makeup of Lassen County was 25,532 (73.2%) White (U.S. Census), **2,834** (8.1%) African American (U.S. Census), 1,234 (3.5%) Native American (U.S. Census), **356** (1.0%) Asian (U.S. Census), 165 (0.5%) Pacific Islander (U.S. Census), 3,562 (10.2%) from Race (United States Census), and 1,212 (3.5%) from two or more races. Hispanic (U.S. Census) or Latino (U.S. Census) of any race were 6,117 persons (17.5%).

Question: How many more people were in the **second biggest** racial group compared to the **second smallest**?

NumNet+: -2834
AeNER: 2478
Answer: **2834** – **356** = 2478

Figure 1: Examples from TAT-QA and DROP with answers produced by NumNet+v3$_{DB}$ (Section 4.3) and AeNER$_{DB}$.

Riedel, 2018; Wallace et al., 2019; Jin et al., 2021; Sundararaman et al., 2020; Thawani et al., 2021; Gorishniy et al., 2022) with the aim of improving numerical MRC. AeNER effectively leverages the number embeddings by passing them through multiple Transformer layers (Vaswani et al., 2017) to obtain a numerically-aware representation of the input question and the context. Compared to the current state-of-the-art approaches (Chen et al., 2020a; Zhou et al., 2022a), our approach is equally applicable to the textual and tabular contexts and does not rely on auxiliary models or heuristically derived rules. Our analysis demonstrates that the proposed approach outperforms the baseline models on the challenging DROP, TAT-QA, and RACENum (Lai et al., 2017) datasets.

Our key contributions are as follows:

- We show that NumNet's reasoning module, a central component of the model that encodes numbers, has little to no effect on the performance with large pre-trained encoders.
- We present AeNER, a simple yet effective model that utilizes numerical representations to improve the model's reasoning ability.
- We evaluate our method on DROP and RACENum with text contexts, and on TAT-QA with hybrid contexts consisting of texts and tables. At the time of submission, our model is ranked #1 on the leaderboards of DROP[2] and TAT-QA[3].

To facilitate further research, we will release our code and model checkpoints.

[2] https://leaderboard.allenai.org/drop/submissions/public
[3] https://nextplusplus.github.io/TAT-QA

2 Related Work

Machine Reading Comprehension Machine reading comprehension is an important area of research in natural language understanding. In recent years, many challenging datasets have emerged (Wang, 2022) and attracted a lot of the community's effort and attention. In SQuAD, a well-known dataset in which the answer is extracted as a span from a text passage, machines have already achieved human-level performance (Rajpurkar et al., 2016). In RACE (Lai et al., 2017) with a multi-choice format, the model must select an answer from four suggested options. HotpotQA (Yang et al., 2018) is a multi-hop QA dataset where the questions require reasoning across multiple documents. Apart from text-only datasets, many datasets include tables as input. In FinQA (Chen et al., 2021), provided with a table and a paragraph, the model is expected to generate an executable program to answer a given question. HybridQA (Chen et al., 2020b) tests the ability to reason over unstructured hybrid data.

Numerical Reasoning over Texts In today's data-driven world, numerical reasoning is a crucial skill. To benchmark it, Dua et al. (2019) collected the DROP dataset with compositional questions involving multiple steps of discrete reasoning. To tackle such questions, they proposed NAQANet, a multi-head predictor method with four "head" output layers for each type of answer. Later, Segal et al. (2020) added a fifth answer type to handle multi-span answers. Andor et al. (2019) developed Bert-Calculator, which outputs a template program that can be executed to produce the answer. NeRd (Chen et al., 2020c) generates an executable pro-

gram in a special domain-specific language to derive the answer. Geva et al. (2020) provided a way to incorporate numerical reasoning skills into a BERT-based model, GenBERT, by training it on a synthetic dataset. NumNet (Ran et al., 2019) and its well-known extension, NumNet+v2, use a graph with numbers as nodes in its reasoning module to create a better representation of numbers in texts.

Among recent state-of-the-art approaches, Chen et al. (2020a) built on NumNet+v2 by adding not only numbers but also related entities to the graph. However, the resulting model, QDGAT, inherently relies on extracting named entities from text and building the graph based on the co-occurrence of numbers and entities in the same sentence, which severely limits the method's applicability to tabular contexts. The newest work by Zhou et al. (2022a) introduced OPERA, an operation-pivoted reasoning mechanism, that leads to better performance but requires an extra training task and heuristic rules for mapping questions from DROP to an ad hoc set of related operations. As of now, there is no working open-source code for either model, which poses implementation and reproducibility challenges due to the complexity of the methods. Our approach is simpler and more scalable in two ways: (1) it works with text, tabular, and hybrid contexts; (2) it does not rely on domain-specific heuristics.

Numerical Reasoning over Hybrid Data Numerical reasoning on hybrid contexts, such as in the TAT-QA dataset, is currently a very active research area. Zhu et al. (2021) proposed the TagOP model, which uses a sequence tagging method to extract a set of tokens as supporting evidence and then predicts a relevant aggregation operator and the answer scale separately. Li et al. (2022) introduced FinMath, a tree-structured neural model that makes predictions based on the extracted pieces of evidence. MHST (Zhu et al., 2022) utilizes the evidences for the arithmetic questions to generate expression trees in a Seq2Tree manner. However, these sequence labeling-based methods show significantly lower performance compared to multi-head predictor approaches, as discussed in Section 4. KIQA (Nararatwong et al., 2022) employs entity-linking models to equip a QA model with external knowledge. Lei et al. (2022) constructed RegHNT, a relational graph-based model that captures the relationship between different data types and then uses a tree-based decoder module to generate an expression tree. Deng et al. (2022) proposed a hybrid Seq2Seq framework, called UniPCQA, for code generation in numerical reasoning tasks. Zhou et al. (2022b) built the UniRPG model that consists of a neural programmer and program executor modules to generate programs in a predefined domain-specific language (DSL). Although program-generating methods have a more interpretable reasoning process, creating a DSL and, possibly, additional program annotations is a laborious and expensive task.

Representing Numbers in NLP Several studies (Naik et al., 2019; Wallace et al., 2019) have shown that numerical embeddings from pre-trained encoders do not fully capture the semantic meaning of the number values. In Section 3.4, we overview various existing methods for encoding numeric tokens. However, this topic is currently still underexplored (Thawani et al., 2021). Sundararaman et al. (2020) presented one of the few works to experiment with number representations in the MRC task. Their method, DICE, shows improvements on the SQuAD dataset but does not help in our setting, as we discuss in Section 4. For the in-depth read on numerical embeddings, we refer the reader to the survey by Thawani et al. (2021).

3 Method

In the machine reading comprehension task, the input is a context P along with a related question Q. The goal is to answer Q based on P. There can be multiple valid answers to each question, and at inference time, a model prediction is considered correct if it matches any valid answer.

The proposed model, AeNER, is structured similarly to previous research (Ran et al., 2019; Chen et al., 2020a; Zhou et al., 2022a) and comprises three components: an encoding module, a reasoning module, and a prediction module. The architecture of AeNER is illustrated in Figure 2.

3.1 Encoding Module

To process the input, we use language representation models as encoders (Devlin et al., 2019; Liu et al., 2019; He et al., 2021a,b). In this work, we focus on two types of context:

- when P is a text passage, the encoder's input is a tokenized concatenation of `[CLS]`, question, `[SEP]`, passage, and `[SEP]`;
- when P is a table with an associated text paragraph, the encoder's input is a tokenized con-

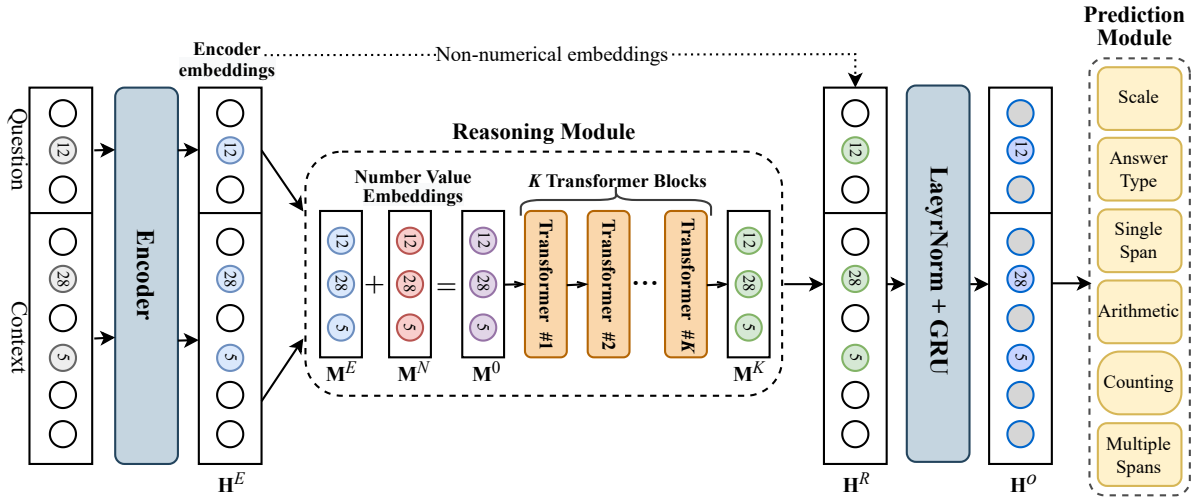

Figure 2: The architecture of AeNER consisting of an encoder, a reasoning module, and a prediction module.

catenation of `[CLS]`, question, `[SEP]`, table, `[SEP]`, passage, and `[SEP]`, where table is flattened by row and the cells are separated with the `[SEP]` token.

The encoder produces a corresponding representation $\mathbf{H}^E = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_l] \in \mathbb{R}^{l \times d}$, where l is the length of the input sequence and d is the dimension of the embedding vectors.

3.2 Reasoning Module

First, we extract the numbers from the input. Let $\mathbf{I}^{(\text{num})} = \{i_1, i_2, \ldots, i_s\}$ be the set of indices of numeric tokens. The encoder embeddings of the numbers can be denoted as $\mathbf{M}^E \in \mathbb{R}^{s \times d}$ and are defined as follows:

$$\mathbf{M}^E_j = \mathbf{h}_{i_j}$$

We perform the element-wise sum of $\mathbf{M}^E$ and the so-called *number value embeddings* $\mathbf{M}^N$:

$$\mathbf{M}^0 = \mathbf{M}^E + \mathbf{M}^N$$

Our approach to obtaining the number value embeddings is described in Section 3.4.

Next, $\mathbf{M}^0$ is passed through K successive Transformer layers (Vaswani et al., 2017), a process we refer to as multi-step Transformer reasoning. Specifically, in the i-th step ($1 \le i \le K$), the output of the previous step is used as input to the i-th Transformer block:

$$\mathbf{M}^i = \text{TransformerLayer}^{(i)}(\mathbf{M}^{i-1})$$

We update the encoder embeddings by replacing the numerical embeddings with $\mathbf{M}^K$. The updated representation is denoted by $\mathbf{H}^R$:

$$\mathbf{H}^R_j = \begin{cases} \mathbf{H}^E_j, & \text{if } j \notin \mathbf{I} \\ \mathbf{M}^K_{I(j)}, & \text{if } j \in \mathbf{I}, \end{cases}$$

where $I(j)$ is the index of the number token in the encoder input such that $i_{I(j)} = j$. The final embeddings are constructed in two steps: first, we apply Layer Normalization (Ba et al., 2016) to $\mathbf{H}^R$, and then we employ a GRU-based (Cho et al., 2014) RNN and take the output hidden states:

$$\mathbf{H}^O = \text{GRU}(\text{LayerNorm}(\mathbf{H}^R))$$

3.3 Prediction Module

To make predictions, we use the obtained embeddings $\mathbf{H}^O$ as the final numerically-aware representation of the question Q and the context P.

The model is trained to maximize the likelihood of the correct answer $\mathcal{A}$ and its scale S by marginalizing over the possible answer types $\mathcal{T}$:

$$p_\theta(\mathcal{A}, S) = \sum_{T \in \mathcal{T}} p_\theta(T) p_\theta(\mathcal{A}|T) p_\theta(S|\mathcal{A}, T)$$

We consider several types of answers and have output layers for each of them, as well as two output layers for classifying the answer type and the scale.

Single Span For span extraction, we model the probabilities of the start and end positions of a span:

$$\mathbf{p}^{\text{start}} = \text{softmax}(\text{FFN}^{\text{start}}(\mathbf{H}^O))$$
$$\mathbf{p}^{\text{end}} = \text{softmax}(\text{FFN}^{\text{end}}(\mathbf{H}^O))$$

Here and later, FFN refers to a single-layer feed-forward neural network. The span probability is computed as the product of the corresponding probabilities in $\mathbf{p}^{\text{start}}$ and $\mathbf{p}^{\text{end}}$. In contrast to existing

research, where the question and context span predictors are separated into different output layers (Dua et al., 2019; Ran et al., 2019; Chen et al., 2020a; Zhou et al., 2022a), we predict spans for the concatenation of the question and the context.

Multi-Spans To extract multiple spans, we follow the sequence tagging method of Segal et al. (2020) where the tokens are labeled with `BIO` tags.

Arithmetic expression Previous works that evaluate on the DROP dataset only allow expressions that include addition and subtraction. To accommodate the operations from TAT-QA, we add three more types of expressions: change ratio, average, and division. Each type of expression can be encoded as a ternary or binary mask of length s. For example, in addition-subtraction expressions, each number can be assigned 0, -1, or 1 depending on its coefficient in the total sum. Let $\mathcal{M}$ be a set of masks corresponding to the expressions that result in a correct answer, then:

$$p_\theta(\mathcal{A}|T) = \sum_{M \in \mathcal{M}} \prod_{j=1}^{s} p_\theta(M_j)$$

The probability of the mask value assigned to a number is modeled as follows:

$$p_\theta(M_j) = \text{softmax}(\text{FFN}(\mathbf{H}^O_{i_j}))$$

Counting Following Dua et al. (2019), we assume that the answer to the counting questions is in the range of 0-9. After mean pooling the embeddings $\mathbf{H}^O$ into a single vector $\mathbf{h}^{\text{pool}}$, we model the counting operation as a 10-class classification task:

$$p_\theta(\mathcal{A}|T) = \text{softmax}(\text{FFN}(\mathbf{h}^{\text{pool}}))$$

Answer Type We model answer type distribution in a similar way to the counting prediction.

Scale To classify the scale type, we compute $\mathbf{h}^{\text{scale}}$ by applying mean pooling only to the tokens that are part of the derived spans or appear in the arithmetic expressions. The distribution is set as:

$$p_\theta(S|\mathcal{A}, T) = \text{softmax}(\text{FFN}(\mathbf{h}^{\text{scale}}))$$

3.4 Number Value Embeddings

In our efforts to construct number value embeddings, we explore various options.

Exponent Embedding This is a simple lookup embedding that depends solely on the exponent of the number (Berg-Kirkpatrick and Spokoyny, 2020). For instance, $114 = 1.14 \times 10^2$ has the exponent of 2. In our implementation, we multiply the lookup embedding by the mantissa part.

DigitCNN Another approach is to view a number as a sequence of digits and utilize CNN or RNN to produce a numerical representation (Spithourakis and Riedel, 2018). In our experiments, we opt for CNN, as it achieves the best overall score on the synthetic tests proposed by Wallace et al. (2019).

Prototype-based Embedding The idea behind prototype-based embeddings (Jin et al., 2021) is to convert the number into scientific notation and then encode the exponent and mantissa part separately.

Periodic Embedding Periodic embeddings originate from the line of research dedicated to deep learning on tabular data. We adopt the approach of Gorishniy et al. (2022) to encode a number x as $[\sin v; \cos v]$ where $v = [2\pi c_1 x, \ldots, 2\pi c_d x]$ and c_i are trainable parameters.

Ranking Embedding Instead of training a model to comprehend numerical values directly, we could show it how the numbers compare to each other. To this end, we propose to sort the numbers from the input by their values and assign each one a rank. To encode a number, we then use a trainable lookup embedding of the number's rank.

Mixed Embedding One question raised by Thawani et al. (2021) in their insightful survey is whether multiple methods to encode numbers can be mix-and-matched. To explore this idea, we concatenate the d-dimensional embeddings listed above and then multiply the result by a projection matrix to produce the vector of dimension d.

DICE Sundararaman et al. (2020) proposed a method to improve the performance of BERT on the SQuAD dataset (Rajpurkar et al., 2016) by using an auxiliary loss for pairs of sampled numbers from the contexts. The loss for (x, y) is computed as $\mathcal{L}_{\text{num}}(x, y) = \left| 2\frac{|x-y|}{|x|+|y|} - d_{\cos}(\mathbf{x}, \mathbf{y}) \right|$ where $\mathbf{x}$ and $\mathbf{y}$ are the corresponding representations of x and y, respectively, and $d_{\cos}(\mathbf{x}, \mathbf{y}) = 1 - \frac{\mathbf{x}^T\mathbf{y}}{\|\mathbf{x}\|_2\|\mathbf{y}\|_2}$ is the cosine distance between $\mathbf{x}$ and $\mathbf{y}$. This approach differs from the above methods as it requires removing the number value embeddings from the AeNER architecture and adding the auxiliary loss to the training objective, instead.

4 Experiments

4.1 Setting

In our experiments, we employ the large versions of RoBERTa (Liu et al., 2019) and DeBERTa-v3 (He

	BLR	LR	Epochs	BS	d
BERT	5e-5	1e-4	25	16	512
RoBERTa	1.5e-5	5e-4	25	16	1024
DeBERTa	1.5e-5	5e-3	25	16	1024

Table 1: Hyperparameter configurations for AeNER.

et al., 2021a,b) as pre-trained encoders. We also conduct experiments with the BERT-base model (Devlin et al., 2019) to evaluate the impact of NumNet+ on a smaller encoder model. All models were trained using the Adam optimizer (Kingma and Ba, 2014) for 25 epochs with a batch size of 16. The hyperparameters used for the DROP dataset are shown in Table 1: BLR, LR, BS, d, and K stand for batch size, encoder learning rate, learning rate of the rest of the network, batch size, and dimension of the output embedding. We perform $K = 3$ Transformer reasoning steps. The only difference in the settings for the TAT-QA dataset is that LR is set to 1e-3 for the DeBERTa-v3 model. On a single A100 GPU, the training required 12 to 48 hours depending on a dataset and encoder.

4.2 Datasets and Evaluation

We use Exact Match (EM) and F1 score as our evaluation metrics. We conduct our experiments on three datasets: DROP (Dua et al., 2019), TAT-QA (Zhu et al., 2021), and RACENum (Lai et al., 2017; Chen et al., 2020a).

DROP DROP is a crowdsourced dataset with passages from Wikipedia articles on history and sports games. The dataset consists of 77,409 questions on 5,565 passages in the training set, 9,536 questions on 582 passages in the development set, and 9,622 questions on 588 passages in the test set.

TAT-QA TAT-QA is composed of hybrid contexts from financial reports and contains 13,215 training set questions, 1,668 development set questions, and 1,669 test set questions on 2,201, 278, and 278 contexts, respectively.

RACENum Following Chen et al. (2020a), we test the generalization properties of AeNER on a subset of the original RACE dataset (Lai et al., 2017) questions that begin with "how many". The data format was modified to conform to the MRC format of DROP. The resulting RACENum dataset includes 1,200 questions, with 629 questions originating from middle school exams (RACE-M) and 571 questions from high school exams (RACE-H).

Method	Dev		Test	
	EM	F1	EM	F1
NAQANet (Dua et al., 2019)	46.20	49.24	44.07	47.01
NumNet (Ran et al., 2019)	64.92	68.31	64.56	67.97
GenBERT (Geva et al., 2020)	68.80	72.30	68.60	72.35
MTMSN_{RB} (Hu et al., 2019)	76.68	80.54	75.85	79.88
$\text{UniRPG}_{\text{BA}}$ (Zhou et al., 2022b)	78.02	80.93	77.08	80.42
NeRd_{RB} (Chen et al., 2020c)	78.55	81.85	78.33	81.71
Calc_{AB} (Andor et al., 2019)	80.22	83.98	79.85	83.56
TASE_{RB} (Segal et al., 2020)	-	-	80.42	83.62
$\text{NumNet+v2}_{\text{RB}}$[4]	81.07	84.42	81.52	84.84
EviDR_{RB} (Zhou et al., 2021)	82.09	85.14	82.55	85.80
QDGAT_{RB} (Chen et al., 2020a)	82.74	85.85	83.23	86.38
$\text{NumNet+v3}_{\text{RB}}$	83.36	85.99	83.39	86.40
OPERA_{RB} (Zhou et al., 2022a)	83.74	86.52	83.55	86.80
OPERA_{AB} (Zhou et al., 2022a)	84.86	87.54	84.69	87.80
Single Model				
AeNER_{RB}	83.72	86.56	83.69	86.98
AeNER_{DB}	**85.58**	**88.32**	**85.78**	**88.88**
Ensemble Model				
AeNER_{DB}	88.31	90.55	87.90	90.33
Human			94.90	96.42

Table 2: Results on the DROP dataset. RB, BA, AB, and DB denote models based on RoBERTa-large, BART-large (Lewis et al., 2020), ALBERT-xxlarge, and DeBERTa-v3-large, respectively.

4.3 Baselines

We outline the main baselines in Section 2. In addition to these, we evaluate a modified version of NumNet+v2, which we call NumNet+v3. In Section 3.3, we specify the two main differences incorporated in our implementation: (1) we use a single span module instead of two separate modules for question and passage spans; (2) to address the operations from TAT-QA, we include three additional arithmetic expressions: change ratio, average, and division. We also compare our results with the Encoder-only model, a simple baseline consisting of only an encoder and a prediction module.

4.4 Main Results

4.4.1 Results on DROP

Table 2 demonstrates the performance of AeNER and other baselines on the DROP dataset. When using the RoBERTa encoder, AeNER performs comparably to OPERA, while outperforming the other models: it exceeds QDGAT (Chen et al., 2020a) by 0.46 EM / 0.6 F1 and NumNet+v3 by 0.3 EM / 0.58 F1. The performance of AeNER_{DB} is significantly superior to that of other baselines. In addition, we train nine AeNER_{DB} models with different random

[4] https://github.com/llamazing/numnet_plus

Method	Dev		Test	
	EM	F1	EM	F1
HyBrider (Chen et al., 2020b)	6.6	8.3	6.3	7.5
BERT-RC (Devlin et al., 2019)	9.5	17.9	9.1	18.7
TaPas (Herzig et al., 2020)	18.9	26.5	16.6	22.8
NumNet+v2$_{RB}$[4]	38.1	48.3	37.0	46.9
TagOp$_{RB}$ (Zhu et al., 2021)	55.2	62.7	50.1	58.0
KIQA$_{RB}$ (Nararatwong et al., 2022)	-	-	58.2	67.4
FinMath$_{RB}$ (Li et al., 2022)	60.5	66.3	58.6	64.1
UniPCQA$_{T5}$ (Deng et al., 2022)	68.2	75.5	63.9	72.2
MHST$_{RB}$ (Zhu et al., 2022)	68.2	76.8	63.6	72.7
UniRPG$_{BA}$ (Zhou et al., 2022b)	70.2	77.9	67.1	76.0
RegHNT$_{RB}$ (Lei et al., 2022)	73.6	81.3	70.3	77.9
Single Model				
AeNER$_{RB}$	75.4	82.2	73.6	82.7
AeNER$_{DB}$	75.8	82.3	-	-
AeNER$_{DB}$-1024	**78.5**	**86.0**	**75.0**	**83.2**
Ensemble Model				
AeNER$_{DB\text{-}1024}$-1024	80.3	86.8	76.8	84.9
Human			84.1	90.8

Table 3: Results on the TAT-QA dataset. RB, BA, T5, and DB denote models based on RoBERTa-large, BART-large (Lewis et al., 2020), CodeT5 (Wang et al., 2021), and DeBERTa-v3-large, respectively. For AeNER$_{DB}$, the hidden test results are to be computed.

Method	BERT		RoBERTa		DeBERTa	
	EM	F1	EM	F1	EM	F1
DROP						
Encoder-only	69.11	72.82	83.31	86.20	84.48	88.09
NumNet+v3	**72.07**	**75.6**	83.36	85.99	84.95	87.85
AeNER	70.49	74.14	**83.72**	**86.56**	**85.58**	**88.32**
TAT-QA						
Encoder-only	57.80	65.47	74.29	81.34	75.48	81.88
NumNet+v3	**59.52**	**67.94**	74.35	81.40	74.70	81.16
AeNER	59.29	67.29	**75.42**	**82.16**	**75.77**	**82.24**

Table 4: Performance of Encoder-only, NumNet+v3, and AeNER on the dev sets of DROP and TAT-QA.

seeds and employ a majority voting strategy to create an ensemble model, resulting in 87.9 EM and 90.33 F1 scores.

4.4.2 Results on TAT-QA

The results for the TAT-QA models are presented in Table 3. Our model, AeNER$_{RB}$ outperforms the baselines by a considerable margin, surpassing the current state-of-the-art RegHNT$_{RB}$ model by 3.3 EM and 4.8 F1. For DeBERTa, which uses relative position embeddings, we increase the maximum sequence length from the default 512 to 1024, since TAT-QA contains a significant number of inputs longer than 512 tokens. The resulting model, AeNER$_{DB}$-1024, achieves a score of 75 EM and 83.2 F1. Ensembling seven of our models further

Method	DROP		TAT-QA	
	EM	F1	EM	F1
Exponent	**85.58**	**88.32**	**75.77**	82.24
DigitCNN	85.29	88.05	74.52	81.57
Prototype-based	85.26	88.04	75.54	82.00
Periodic	85.42	88.06	75.24	82.03
Ranking	85.51	88.19	75.36	**82.50**
Mixed	85.48	88.07	74.58	81.38
DICE: Encoder + $\mathcal{L}_{num}$	85.37	88.03	75.00	82.35

Table 5: Performance of DeBERTa-based AeNER with different number value embeddings.

improves the performance.

In TAT-QA, the annotations for arithmetic and counting questions include gold derivations. We utilize them in the prediction module to identify the correct arithmetic expressions as detailed in Section 3.3. Interestingly, we find that although this additional supervision significantly reduces the number of considered expressions, it has little impact on the performance of AeNER. It is also worth noting that AeNER's answers to arithmetic questions are interpretable, as they come with predicted expressions. To enhance interpretability further, we experiment with training the counting output module to predict the set of counted items from the gold derivations, similar to the approach used for multi-spans, and return their number. This approach results in a slightly lower performance of 74.88 EM / 81.93 F1 on the development set when using DeBERTa, but offers better interpretability.

4.4.3 Comparison with Encoder-only baseline

In Table 4, we compare the performance of NumNet+v3 and AeNER with the Encoder-only model, which posses no reasoning module. The impact of NumNet's reasoning module is evident when using the BERT-base encoder: its advantage over Encoder-only is 2.96 EM / 2.78 F1 on DROP and 1.72 EM / 2.47 F1 on TAT-QA. However, for RoBERTa and DeBERTa encoders, there is no significant difference in performance between NumNet+v3 and Encoder-only. By contrast, the AeNER model consistently outperforms the Encoder-only model even when using larger encoders. The improvement of RoBERTa-based AeNER over Encoder-only is 0.82 F1 on DROP and 0.36 F1 on TAT-QA. For DeBERTa-based AeNER, the advantage is 0.36 F1 on DROP and 0.23 F1 on TAT-QA.

Method	DROP		TAT-QA	
	EM	F1	EM	F1
RoBERTa-based				
AeNER_{RB}	**83.72**	**86.56**	**75.42**	**82.16**
w/o Transformer Reasoning	82.61	85.64	74.46	81.76
w/o Numerical Embeddings	83.18	86.11	74.40	81.33
DeBERTa-based				
AeNER_{DB}	**85.58**	**88.32**	75.77	82.24
w/o Transformer Reasoning	85.21	88.13	**75.83**	**82.35**
w/o Numerical Embeddings	85.05	87.91	74.88	81.61

Table 6: Ablation Study for DeBERTa- and RoBERTa-based AeNER on the dev sets of DROP and TAT-QA.

4.5 Analysis of Number Value Embeddings

We examine several number value embeddings described in Section 3.4. We use the code from Wallace et al. (2019)[5] for DigitCNN and the implementation of Gorishniy et al. (2022)[6] for the periodic embeddings. For the other approaches, we use our own implementation. For the DICE method, we follow Sundararaman et al. (2020) and perform a hyperparameter search for the regularization parameter λ, ultimately settling on $\lambda = 0.025$. As indicated in Table 5, exponent embeddings exhibit a slight advantage in performance compared to the other methods, with an average margin of 0.19 EM / 0.24 F1 on DROP and 0.72 EM / 0.34 F1 on TAT-QA. Therefore, we use exponent embeddings in our default setting. We find that the mixed embeddings generally perform poorly, particularly on the TAT-QA task. Our attempts to utilize more compact parameterizations for mixed embeddings also yielded negative results.

4.6 Ablation Study

We conduct an ablation analysis to assess the effect of the different components of AeNER. The results presented in Table 6 reveal that when using the RoBERTa encoder, removing the Transformer Reasoning mechanism leads to a performance drop of 1.21 EM / 0.92 F1 on DROP and 0.96 EM / 0.4 F1 on TAT-QA. For the DeBERTa encoder, the performance degradation is 0.37 EM / 0.19 F1 on DROP but there is no significant change on TAT-QA. Our observations suggest that the Transformer Reasoning mechanism generally improves the performance of AeNER and makes the use of numerical embeddings more effective.

When ablating number value embeddings, the performance of AeNER_{RB} decreases by 0.54 EM / 0.45 F1 on DROP and 1.02 EM / 0.83 F1 on TAT-QA. For DeBERTa-based AeNER_{DB}, the performance decrease is 0.53 EM / 0.41 F1 on DROP and 0.89 EM / 0.63 F1 on TAT-QA. The results indicate that incorporating numerical embeddings enhances the model's reasoning capabilities.

Method	RACE-M		RACE-H		Overall	
	EM	F1	EM	F1	EM	F1
NumNet+v3	41.71	41.82	29.73	29.73	35.94	36.00
QDGAT	44.01	44.01	28.85	28.85	36.71	36.71
OPERA	47.62	47.62	32.21	32.30	40.20	40.24
AeNER_{RB}	**49.60**	**50.45**	**33.80**	**34.31**	**42.08**	**42.77**

Table 7: Results of RoBERTa-based models on RACENum in the zero-shot setting.

4.7 Performance on RACENum

We compare the generalization performance of the AeNER model with OPERA, QDGAT, and NumNet+v3 on the RACENum dataset. In this setting, the models trained on DROP are asked to perform inference on RACENum in a zero-shot regime. We compare the models based on the same RoBERTa-large encoder. The results shown in Table 7 indicate that AeNER outperforms the other models, achieving a significant improvement of 1.88 EM and 2.53 F1 compared to the second-best performing model, OPERA. This suggests that AeNER has superior generalizability to new domains.

5 Conclusion

We present AeNER, a novel method for numerical reasoning over texts and tables. In our experiments, we investigate different ways to create numerical embeddings. We show that our method outperforms other strong baseline models on the DROP, RACENum, and TAT-QA datasets. Our ablation analysis indicates that the number value embeddings and the Transformer Reasoning mechanism are critical components of the model. AeNER demonstrates strong numerical reasoning capabilities and can be easily trained on both text and tabular data without significant domain-specific adjustments. However, our approach still requires task-specific output layers in the prediction module. In the future, we plan to explore the potential of our approach for generative reasoning models.

[5] https://github.com/Eric-Wallace/numeracy

[6] https://github.com/Yura52/tabular-dl-num-embeddings

References

Daniel Andor, Luheng He, Kenton Lee, and Emily Pitler. 2019. Giving BERT a calculator: Finding operations and arguments with reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5947–5952, Hong Kong, China. Association for Computational Linguistics.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization.

Taylor Berg-Kirkpatrick and Daniel Spokoyny. 2020. An empirical investigation of contextualized number prediction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4754–4764, Online. Association for Computational Linguistics.

Kunlong Chen, Weidi Xu, Xingyi Cheng, Zou Xiaochuan, Yuyu Zhang, Le Song, Taifeng Wang, Yuan Qi, and Wei Chu. 2020a. Question directed graph attention network for numerical reasoning over text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6759–6768, Online. Association for Computational Linguistics.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020b. HybridQA: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1026–1036, Online. Association for Computational Linguistics.

Xinyun Chen, Chen Liang, Adams Wei Yu, Denny Zhou, Dawn Song, and Quoc V. Le. 2020c. Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. 2021. FinQA: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Yang Deng, Wenqiang Lei, Wenxuan Zhang, Wai Lam, and Tat-Seng Chua. 2022. Pacific: Towards proactive conversational question answering over tabular and textual data in finance.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

James Ferguson, Matt Gardner, Hannaneh Hajishirzi, Tushar Khot, and Pradeep Dasigi. 2020. IIRC: A dataset of incomplete information reading comprehension questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1137–1147, Online. Association for Computational Linguistics.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 946–958, Online. Association for Computational Linguistics.

Yury Gorishniy, Ivan Rubachev, and Artem Babenko. 2022. On embeddings for numerical features in tabular deep learning. *arXiv*, 2203.05556.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021a. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021b. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online. Association for Computational Linguistics.

Minghao Hu, Yuxing Peng, Zhen Huang, and Dongsheng Li. 2019. A multi-type multi-span network

for reading comprehension that requires discrete reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1596–1606, Hong Kong, China. Association for Computational Linguistics.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577, Hong Kong, China. Association for Computational Linguistics.

Zhihua Jin, Xin Jiang, Xingbo Wang, Qun Liu, Yong Wang, Xiaozhe Ren, and Huamin Qu. 2021. Numgpt: Improving numeracy ability of generative pre-trained models. *CoRR*, abs/2109.03137.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Fangyu Lei, Shizhu He, Xiang Li, Jun Zhao, and Kang Liu. 2022. Answering numerical reasoning questions in table-text hybrid contents with graph-based encoder and tree-based decoder. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1379–1390, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

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, Online. Association for Computational Linguistics.

Chenying Li, Wenbo Ye, and Yilun Zhao. 2022. FinMath: Injecting a tree-structured solver for question answering over financial reports. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 6147–6152, Marseille, France. European Language Resources Association.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv preprint*, abs/1907.11692.

Swaroop Mishra, Arindam Mitra, Neeraj Varshney, Bhavdeep Sachdeva, Peter Clark, Chitta Baral, and Ashwin Kalyan. 2022. NumGLUE: A suite of fundamental yet challenging mathematical reasoning tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3505–3523, Dublin, Ireland. Association for Computational Linguistics.

Aakanksha Naik, Abhilasha Ravichander, Carolyn Rose, and Eduard Hovy. 2019. Exploring numeracy in word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3374–3380, Florence, Italy. Association for Computational Linguistics.

Rungsiman Nararatwong, Natthawut Kertkeidkachorn, and Ryutaro Ichise. 2022. KIQA: Knowledge-infused question answering model for financial table-text data. In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 53–61, Dublin, Ireland and Online. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Qiu Ran, Yankai Lin, Peng Li, Jie Zhou, and Zhiyuan Liu. 2019. NumNet: Machine reading comprehension with numerical reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2474–2484, Hong Kong, China. Association for Computational Linguistics.

Elad Segal, Avia Efrat, Mor Shoham, Amir Globerson, and Jonathan Berant. 2020. A simple and effective model for answering multi-span questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3074–3080, Online. Association for Computational Linguistics.

Georgios Spithourakis and Sebastian Riedel. 2018. Numeracy for language models: Evaluating and improving their ability to predict numbers. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2104–2115, Melbourne, Australia. Association for Computational Linguistics.

Dhanasekar Sundararaman, Shijing Si, Vivek Subramanian, Guoyin Wang, Devamanyu Hazarika, and Lawrence Carin. 2020. Methods for numeracy-preserving word embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4742–4753, Online. Association for Computational Linguistics.

Avijit Thawani, Jay Pujara, Filip Ilievski, and Pedro Szekely. 2021. Representing numbers in NLP: a survey and a vision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–656, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP models know numbers? probing numeracy in embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5307–5315, Hong Kong, China. Association for Computational Linguistics.

Yue Wang, Weishi Wang, Shafiq Joty, and Steven C.H. Hoi. 2021. CodeT5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8696–8708, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhen Wang. 2022. Modern question answering datasets and benchmarks: A survey.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Xikun Zhang, Deepak Ramachandran, Ian Tenney, Yanai Elazar, and Dan Roth. 2020. Do language embeddings capture scales? In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4889–4896, Online. Association for Computational Linguistics.

Yongwei Zhou, Junwei Bao, Chaoqun Duan, Haipeng Sun, Jiahui Liang, Yifan Wang, Jing Zhao, Youzheng Wu, Xiaodong He, and Tiejun Zhao. 2022a. OPERA: Operation-pivoted discrete reasoning over text. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1655–1666, Seattle, United States. Association for Computational Linguistics.

Yongwei Zhou, Junwei Bao, Chaoqun Duan, Youzheng Wu, Xiaodong He, and Tiejun Zhao. 2022b. Unirpg: Unified discrete reasoning over table and text as program generation.

Yongwei Zhou, Junwei Bao, Haipeng Sun, Jiahui Liang, Youzheng Wu, Xiaodong He, Bowen Zhou, and Tiejun Zhao. 2021. Evidr: Evidence-emphasized discrete reasoning for reasoning machine reading comprehension. In *Natural Language Processing and Chinese Computing*, pages 439–452, Cham. Springer International Publishing.

Fengbin Zhu, Wenqiang Lei, Fuli Feng, Chao Wang, Haozhou Zhang, and Tat-Seng Chua. 2022. Towards complex document understanding by discrete reasoning.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance. 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 3277–3287, Online. Association for Computational Linguistics.