PASTA: Table-Operations Aware Fact Verification via Sentence-Table Cloze Pre-training

Zihui Gu[1], **Ju Fan**[1], **Nan Tang**[2], **Preslav Nakov**[3], **Xiaoman Zhao**[1*], **Xiaoyong Du**[1]
[1]Renmin University of China, Beijing, China,
[2]Qatar Computing Research Institute, HBKU, Doha, Qatar
[3]Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE
[1]`{guzh,fanj,xiaomanzhao,duyong}@ruc.edu.cn`,
[2]`ntang@hbku.edu.qa`, [3]`preslav.nakov@mbzuai.ac.ae`

Abstract

Fact verification has attracted a lot of research attention recently, e.g., in journalism, marketing, and policymaking, as misinformation and disinformation online can sway one's opinion and affect one's actions. While fact-checking is a hard task in general, in many cases, false statements can be easily debunked based on analytics over tables with reliable information. Hence, table-based fact verification has recently emerged as an important and growing research area. Yet, progress has been limited due to the lack of datasets that can be used to pre-train language models (LMs) to be aware of common table operations, such as aggregating a column or comparing tuples. To bridge this gap, in this paper we introduce PASTA, a novel state-of-the-art framework for table-based fact verification via pre-training with synthesized sentence–table cloze questions. In particular, we design six types of common sentence–table cloze tasks, including `Filter`, `Aggregation`, `Superlative`, `Comparative`, `Ordinal`, and `Unique`, based on which we synthesize a large corpus consisting of 1.2 million sentence–table pairs from WikiTables. PASTA uses a recent pre-trained LM, DeBERTaV3, and further pre-trains it on our corpus. Our experimental results show that PASTA achieves new state-of-the-art performance on two table-based fact verification benchmarks: TabFact and SEM-TAB-FACTS. In particular, on the complex set of TabFact, which contains multiple operations, PASTA largely outperforms the previous state of the art by **4.7** points (85.6% vs. 80.9%), and the gap between PASTA and human performance on the small TabFact test set is narrowed to just **1.5** points (90.6% vs. 92.1%). [1]

1 Introduction

Fact verification, which checks the factuality of a statement, is crucial for journalism (Shu et al., 2017), and is increasingly being applied in other fields (Ott et al., 2011; Yoon et al., 2019). According to Duke Reporters' Lab, there are 300+ active certified fact-checking organizations worldwide.[2]

Automatic and explainable approaches, a.k.a. reference-based approaches, are widely used to assist fact-checkers. They verify the input statement against a trusted source, such as relevant passages from Wikipedia (Popat et al., 2017; Thorne et al., 2018; Shaar et al., 2020). Recently, table-based fact verification has been extensively studied (Chen et al., 2020a; Zhong et al., 2020; Eisenschlos et al., 2020) due to the wide availability of tabular data.

Evidently, performing fact verification over tables requires the ability to reason about **table-based operations**, such as aggregating the values in a column or comparing tuples. For example, for the statement S_1 in Figure 1, it is desirable to reason about the operation over table T that compares the viewers of `Night Moves` with 3.61 to determine whether S_1 is *entailed* or *refuted* by T.

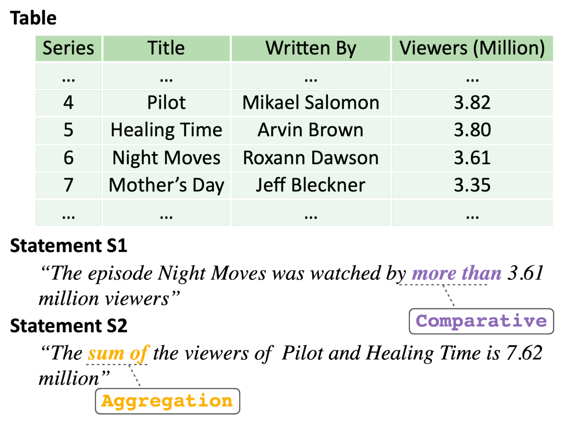
Most previous work (Herzig et al., 2020; Wang et al., 2021a; Schlichtkrull et al., 2021) leverages pre-trained language models (LMs) (Devlin et al., 2019; Liu et al., 2019), which are originally designed for unstructured data, and have a key limitation of overlooking such operations. Some approaches (Zhong et al., 2020; Yang et al., 2020) attempt to explicitly capture the operations by generating a logical form (e.g., a tree) containing the operations from the statement via semantic parsing techniques. However, such approaches face the problem of "spurious programs" (Chen et al., 2020a), due to weak supervision signals in semantic parsing.

To address the above issues, we propose PASTA, a table-operations aware approach. Instead of relying on semantic parsing, PASTA captures table-based operations by designing a novel **sentence–**

*Xiaoman Zhao is the corresponding author.

[1]The pre-trained model, the pre-training corpus, and the source code are released at `https://github.com/ruc-datalab/PASTA`

[2]`http://reporterslab.org/fact-checking-count-tops-300-for-the-first-time/`



Table

Series	Title	Written By	Viewers (Million)
...	...	...	...
4	Pilot	Mikael Salomon	3.82
5	Healing Time	Arvin Brown	3.80
6	Night Moves	Roxann Dawson	3.61
7	Mother's Day	Jeff Bleckner	3.35
...	...	...	...

Statement S1

"The episode Night Moves was watched by more than 3.61 million viewers"

Statement S2

"The sum of the viewers of Pilot and Healing Time is 7.62 million"

Comparative
Aggregation

Figure 1: An example of table-based fact verification.

table cloze pre-training strategy that better guides LMs to reason about table-based operations.

We tackle two challenges for pre-training LMs towards supporting table-based fact verification:

- **Challenge 1:** What types of tasks should be designed, so as to pre-train (or teach) LMs to be aware of operations over tables?

- **Challenge 2:** How to obtain a large-scale and high-quality corpus for pre-training?

To address **Challenge 1**, PASTA automatically synthesizes sentence–table cloze questions. It first synthesizes operations from tables, and then generates cloze tasks by masking the key tokens corresponding to the table-based operations, e.g., "*more than*" and "*sum of*" in Figure 1. Then, LMs are pre-trained to predict the masked operation-aware tokens based on the tables.

Regarding **Challenge 2**, PASTA uses a large table collection, WikiTables (Bhagavatula et al., 2013), and for each table, it synthesizes a diverse set of cloze tasks with six types of table-based operations, including `Filter`, `Aggregation`, `Superlative`, `Comparative`, `Ordinal`, and `Unique`.

For implementation, PASTA uses a recent pre-trained LM, DeBERTaV3 (He et al., 2021a,b) with better positional encoding of the input. To cope with the limited input length of DeBERTaV3, we introduce a select-then-rank strategy for large tables, which further improves the performance.

In sum, we make the following contributions.

- We propose PASTA, a table-operations aware fact verification approach without explicitly generating logical forms.

- We propose a new benchmark for pre-training LMs to be aware of common table-based operations by automatically synthesizing sentence-table cloze questions from WikiTables. In particular, we synthesize a large corpus consisting of 1.2 million sentence-table cloze questions, which we release for future research.

- We evaluate PASTA, which is DeBERTaV3 pre-trained with our table-operations aware pre-training approach, on two widely-adopted table-based fact verification benchmark datasets, TabFact (Chen et al., 2020a) and SEM-TAB-FACTS (Wang et al., 2021b). The experimental results show that PASTA achieves new state-of-the-art (SOTA) results on the two datasets. In particular, on the complex set of TabFact that contains multiple operations, PASTA outperforms the previous SOTA by **4.7** points (85.6% vs. 80.9%), and the gap between PASTA and human performance on the small test set is narrowed to **1.5** points (90.6% vs. 92.1%).

2 Preliminaries

2.1 Problem Formulation

Let T be a table with m columns and n rows. Let $T_{i,j}$ denote the cell in the i-th column and j-the row of T. Let S be a natural language (`NL`) statement.

The problem of **table-based fact verification** is formulated as follows: Given an `NL` statement S and a table T, it determines whether statement S can be *entailed* or *refuted* by table T[3].

See Figure 1 for an example table about movies and their viewers, and two statements where S_1 contains a `Comparative` operation and S_2 contains an `Aggregation` operation.

2.2 DeBERTa for Sentence-Table Encoding

Inspired by the success of BERT-like models (Devlin et al., 2019; Liu et al., 2019; Clark et al., 2020) in natural language understanding (NLU) tasks, many existing studies leverage pre-trained LMs for table understanding, achieving superior results (Chen et al., 2020b; Schlichtkrull et al., 2021). In this paper, we apply DeBERTa (He et al., 2021b) for sentence-table encoding, as it can effectively capture positional information of the input with its

[3]This formulation is in line with TabFact (Chen et al., 2020a). Note that other variants of this problem are also possible, e.g., with a third option *not sure*.

positional encoding scheme, which is useful for sentence-table encoding.

Given an input token at position i, DeBERTa represents it using two vectors, $\{H_i\}$ and $\{P_{i|j}\}$, to represent its content and relative position with respect to the token at position j. For a single-head self-attention layer, DeBERTa (He et al., 2021b) represents the disentangled self-attention mechanism as follows:

$$Q_c = HW_{q,c}, K_c = HW_{k,c}, V_c = HW_{v,c}$$
$$Q_r = PW_{q,r}, K_r = PW_{k,r}$$
$$\tilde{A}_{i,j} = Q_i^c {K_j^c}^{\mathrm{T}} + Q_i^c {K_{\delta(i,j)}^r}^{\mathrm{T}} + K_j^c {Q_{\delta(j,i)}^r}^{\mathrm{T}},$$

where $\tilde{A}_{i,j}$ represents the attention score between token i and token j. The content vector H is projected by the matrices $W_{q,c}, W_{k,c}, W_{v,c} \in R^{d\times d}$ to generate the projected content vectors Q_c, K_c and V_c, respectively, $P \in R^{2k\times d}$ is the relative position embedding vector, and $\delta(i,j)$ is the relative distance from token i to token j. Similarly to H, P is projected by the matrices $W_{q,r}, W_{k,r} \in R^{d\times d}$ to generate the projected position vectors Q_r and K_r, respectively.

In our implementation, we adopt the latest version DeBERTaV3 (He et al., 2021a), which improves DeBERTa by further pre-training with replaced token detection (RTD) to jointly encode a sentence and a table. Unlike `NL`, tables have distinct structural information that is difficult to capture by a pre-trained LMs. Therefore, we use special symbols to inject the structural information into an `NL` sentence. Specifically, we linearize the table $T = \{T_{i,j}|i \le m, j \le n\}$ into sentence S_T = [`Header`] $T_{0,0}$ | $T_{0,1}$ ... | $T_{0,n}$ [`Row`] $T_{1,0}$ | $T_{1,1}$ ... [`Row`] $T_{i,0}$ | $T_{i,1}$ ... | $T_{m,n}$. We use [`Header`] to indicate the beginning of the headers and [`Row`] to indicate the beginning of each row. Inspired by Liu et al., 2021, we also use "|" to separate each cell. Afterwards, we concatenate the statement S and the linearized table S_T.

3 Our PASTA Model

Figure 2 gives an overview of our PASTA framework, which follows the pre-training–fine-tuning framework. For **pre-training**, we guide our model to understand sentences and to perform table-based operations (e.g., `Aggregation`) to complete ***synthesized cloze tasks*** in the sentence. For **fine-tuning**, we apply a ***select-then-rank*** strategy to trade off between sizes of large tables and the limited input length of DeBERTaV3. Next, we will first present our sentence-table cloze task and pre-training corpus generation in Sections 3.1 and 3.2, respectively. We will then discuss our fine-tuning strategy in Section 3.3.

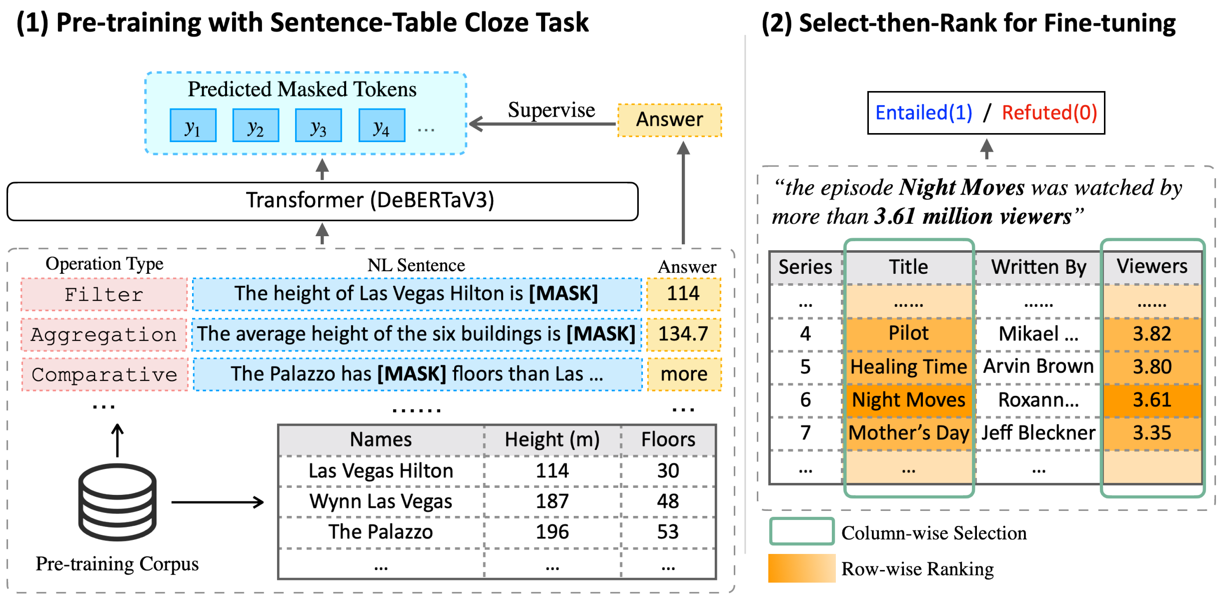
3.1 Sentence-Table Cloze Pre-training

An essential ability for our fact verification model is to be capable of reasoning about **table-based operations** on the entities of the table. Although it is difficult to enumerate all the expressions of real-world statements, the types of operations (e.g., `Unique`) expressed by statements could be limited. Therefore, it is possible and reasonable for the model to understand these operations. To this end, we propose a table-operations aware pre-training task which can guide the model to understand table-aware operations expressed by the statements.

Inspired by the Masked Language Modeling (MLM) (Devlin et al., 2019), we design a **cloze task** to pre-train the model's ability to reason about operations over tables. However, the key difference is that we do not use the *random masking* strategy in MLM due to the following reasons. First, masking a specific cell of the table and training the model to predict it is difficult, because, unlike the words in a sentence, the contents of a cell may not be predictable from the contents of the surrounding cells. Moreover, the content of an individual cell may be useless for determining whether statement S can be entailed or refuted by table T. Second, not every token in the sentence needs to be predicted. For example, in the sentence *"The Palazzo has more floors than Las Vegas Hilton."*, tokens like *has* and *the*, which can be easily predicted from contextual information, are not worth learning for the model because this kind of ability is already captured by pre-trained LMs.

To pre-train the model to be aware of table operations, we propose to **mask operation-aware tokens** in the sentence, which need to meet two requirements: (*i*) to appear in the sentence and to correspond to table-based operations, and (*ii*) to be predictable by reasoning over tables. For example, in the above sentence, as the model needs to find the numbers of floors for "*Las Vegas Hilton*" and for "*The Palazzo*" in the table and then to compare them, "*more*" is the operation-aware token that the model needs to predict.

To cover common types of operations in pre-training, we refer to the operations list defined in LPA (Chen et al., 2020a). We design **six sentence-**


(1) Pre-training with Sentence-Table Cloze Task
Predicted Masked Tokens
y_1 y_2 y_3 y_4 ...
Supervise
Answer
Transformer (DeBERTaV3)
Pre-training Corpus
(2) Select-then-Rank for Fine-tuning
Entailed(1) / Refuted(0)
"the episode Night Moves was watched by more than 3.61 million viewers"
Column-wise Selection
Row-wise Ranking

Operation Type	NL Sentence	Answer
`Filter`	The height of Las Vegas Hilton is **[MASK]**	114
`Aggregation`	The average height of the six buildings is **[MASK]**	134.7
`Comparative`	The Palazzo has **[MASK]** floors than Las ...	more
...	......	...

Names	Height (m)	Floors
Las Vegas Hilton	114	30
Wynn Las Vegas	187	48
The Palazzo	196	53
...	...	...

Series	Title	Written By	Viewers
...	......	......	......
4	Pilot	Mikael ...	3.82
5	Healing Time	Arvin Brown	3.80
6	Night Moves	Roxann...	3.61
7	Mother's Day	Jeff Bleckner	3.35
...	...	...	

Figure 2: An overview of the pre-training and fine-tuning procedures of PASTA.

table cloze tasks according to various operation types: `Filter`, `Aggregation`, `Superlative`, `Comparative`, `Ordinal`, and `Unique`. Figure 2 shows some example cloze tasks of operation types, `Filter`, `Aggregation` and `Comparative`. The answer to a cloze task, i.e., operation-aware tokens to be predicted, may be a specific table cell (e.g., "*114*") or the result of a series of calculations (e.g., "*134.7*", "*more*"). Note that we assume that only atomic operation types need to be learned in pre-training, and various combinations and expressions are left to fine-tuning. Thus, only one type of operation-aware token is masked in each statement.

We formally define the table-operations aware pre-training task as follows: Given a sentence $S = \{x_i\}$ and a table $T = \{T_{i,j} | i \leq m, j \leq n\}$, we corrupt S into $\tilde{S}$ by masking the operation-aware span of tokens $S_{span} = \{\tilde{x}_i\} \subset S$ in it, and then we train an LM parameterized by θ to reconstruct S by predicting the masked tokens $\{\tilde{x}_i\}$, i.e., optimizing the following objective:

$$\begin{aligned} L_{\text{PASTA}} &= -\log p_\theta(S|\tilde{S}, T) \\ &= -\sum_{\tilde{x}_i \in S_{span}} \log p_\theta(\tilde{x}_i = x_i | \tilde{S}, T) \end{aligned}$$

3.2 Pre-training Corpus Generation

Next, we introduce our strategy for generating the pre-training corpus consisting of sentence–table pairs. According to the table-operations aware pre-training task described in Section 3.1, the difficulty of corpus generation is how to collect a large scale of sentence–table pairs and how to identify the operation-aware tokens in each sentence. To solve these problems, we propose an automatic data generation method, which consists of table collection and sentence generation. In addition, we also introduce a probing-based sentence polishing method to make it more fluent and natural.

Table Collection. Inspired by previous work (Herzig et al., 2020; Schlichtkrull et al., 2021), we use WikiTables,[4] which contains Web tables extracted from Wikipedia. Concretely, we only select well-formed relational tables that contain headers and at least one numeric column that can be used for operations. Moreover, considering the maximum input length (512 tokens) of our pre-trained LM, we filter out all tables with more than 500 cells. Based on the above process, we obtain a total of 580K tables from WikiTables, and we then randomly select 20K tables to improve the efficiency of pre-training.

Sentence Generation. Figure 3 shows the pipeline of our automatic sentence generation method. To ensure that each sentence contains an operation and the operation-aware tokens can be clearly identified, we design `NL Templates` for each table-aware operation type (e.g., the `NL Template` in Figure 3 is designed for `Comparative`. See more details of the manually designed templates in Appendix A). Each `NL Template` is pre-defined with the position of the operation-aware tokens (e.g., "*higher*").

[4] `http://websail-fe.cs.northwestern.edu/TabEL/`

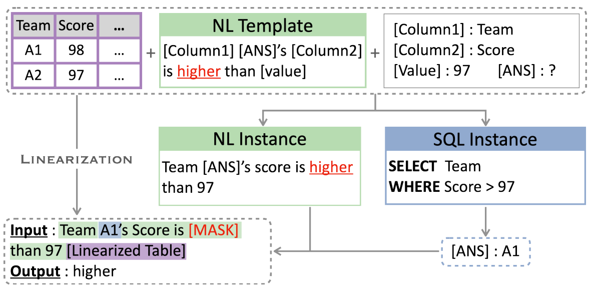


Figure 3: Automatic pre-training corpus generation.

Type	# Sentence-Table	Len (Ans)
`Filter`	77,609 (6%)	3.2
`Superlative`	349,241 (27%)	2.6
`Aggregation`	388,046 (30%)	1.3
`Comparative`	349,241 (27%)	1.0
`Ordinal`	103,479 (8%)	2.3
`Unique`	25,872 (2%)	1.0
Total	1,293,488	1.8

Table 1: Statistics of our pre-training corpus, where "Len (Ans)" represents the average length of the answer.

Unlike fact verification, the sentence in the cloze task must be a correct description of the table. To achieve this, we design an `SQL Template` for each `NL Template`, and these two templates will be instantiated based on the table at the same time. During instantiation, the [`Column`] in both templates is replaced by a column header (e.g., [`Column1`] is replaced by `team`) and the [`Value`] is instantiated by a cell (e.g., [`Value`] is replaced by `97`). Then, the `SQL Instance` will be automatically executed on the table, and the execution result [`ANS`] will be filled in the `NL Instance` to ensure its correctness.

Based on the above method, we generate up to 100 related sentences for each table, depending on the size of the table. Statistics about the pre-training corpus are given in Table 1. We can see that the proportion of each type is different; it mainly depends on how many different expressions a type contains. Taking `Aggregation` with the highest proportion as an example: in addition to *"the average of"* in Figure 2, there may also be *"the sum of"*, *"the total amount of"*, etc.

Sentence Polishing. We notice that using fixed templates for each table could generate unnatural sentences. For example, in Figure 3, if [`Column2`] is not populated with *"score"* but by *"age"*, then the operation-aware token should use *"older"* instead of *"higher"*. Therefore, we introduce a probing-based method to improve the fluency of sentences, which leverages the rich knowledge learned by the LMs implicitly during pre-training. Our main idea is that since BERT-like LMs are pre-trained on extensive textual corpora, their predictions can approximate the natural language expressions used in real-world scenarios.

Specifically, we identify the context sensitive word w' in each template (e.g., *"higher"*), and define a set of candidate values (e.g.,*"higher"*, *"more"*, . . . , *"older"*) for w'. Then, we replace the w' with [`MASK`] and leverage a fixed LM to determine the appropriate value for [`MASK`] based on context. For example, if [`Column2`] is populated as *"age"*, the pre-trained LM calculates probabilities for all candidate values and then selects the one with the highest probability, e.g., *"older"*. Please refer to Appendix A for the detailed definition of context sensitive words and their candidate sets.

3.3 Fine-tuning with Select-then-Rank

For fine-tuning, we pre-process the table based on the following two considerations: (i) As mentioned in Eisenschlos et al., 2020, the sentence-table pairs in the downstream datasets may be too long for pre-trained LMs, and (ii) considering that the disentangled attention mechanism in DeBERTa makes the model more sensitive to the positional information of the input, we assume that it would be easier for the model to capture the sentence-table relationship by putting the most relevant cells in the table closer to the sentence. To address the above two problems, we propose a **select-then-rank** method to reconstruct the table content.

Column-wise Selection. To make the table size to meet the input length limit of DeBERTaV3, we follow previous work (Chen et al., 2020a) to only select columns in the table containing entities linked to the statement, which results in a pre-processed table $\tilde{T}$. Note that the reason for not selecting by rows is that some operations may involve an entire column of cells, e.g., `Aggregation`.

Row-wise Ranking. To make the sentence and its relevant cells in the table have closer positions, we reorder the table $\tilde{T}$ by row. Specifically, we slice $\tilde{T}$ into a set of rows $\{r_1, \ldots, r_m\}$, and rank these rows by their relevance scores $\{p_i\}$, as defined below. Let $\hat{r}_i$ and $\hat{s}$ denote the token sets of row r_i and statement s respectively. The relevance score p_i is given by $|\hat{r}_i \cap \hat{s}|$. Note that we remove the stopwords (e.g., *the*) in $\hat{r}_i$ and $\hat{s}$. Finally, we reconstruct the table by ordering the rows in descending order of the relevant scores, before applying the table linearization introduced in Section 2.2.

4 Experimental Setup

4.1 Datasets

By using the method introduced in 3.2, we synthesize 1.2 million sentence-table cloze questions as the pre-training corpus. Specifically, our pre-training corpus contains 20K well-structured tables selected from WikiTables, and sentences are automatically generated from the tables.

During fine-tuning, we evaluate our model on two widely-adopted table-based fact verification benchmark datasets TabFact (Chen et al., 2020a) and SEM-TAB-FACTS (Wang et al., 2021b). **TabFact** contains 16K tables collected from WikiTables and 118K human-annotated natural language statements, where each statement-table pair is labeled as either *entailed* or *refuted*. TabFact contains statements with two difficulty levels: (i) simple statements corresponding to single rows, and (ii) complex statements involving multiple rows with table-based operations like `Aggregation`. **SEM-TAB-FACTS** contains 2K tables and 4K human-annotated natural language statements. Different from TabFact, these tables are collected from scientific articles in a variety of domains. We use the official splits in the two benchmarks for evaluation: the training, validation and test sets of TabFact respectively contain 92283, 12792 and 12779 sentence-table pairs; the training, validation and test sets of SEM-TAB-FACTS respectively contain 4506, 423 and 522 sentence-table pairs. In addition, TabFact also holds out a small test set with 2K sentence-table pairs with *human performance*.

4.2 Baselines

We evaluate PASTA with the following ten state-of-the-art methods for table-based fact verification.

Table-BERT (Chen et al., 2020a) adopts templates to linearize a table into an NL sentence, and then directly leverages a BERT model to encode the linearized table and the statement.

LogicFactChecker (Zhong et al., 2020) leverages a sequence-to-action semantic parser to generate a "program", i.e., a tree with multiple operations, and uses a graph neural network to encode statements, tables, and the generated programs.

SAT (Zhang et al., 2020) creates a structure-aware mask matrix to encode the structural data. In particular, it considers recovering the alignment information of tabular data by masking signals of unimportant cells during self-attention.

ProgVGAT (Yang et al., 2020) integrates programs and execution into a natural language inference model. This method uses a verbalization with a program execution model to accumulate evidences and constructs a graph attention network to combine various evidences.

Tapas (Herzig et al., 2020) extends BERT with additional structure-aware positional embeddings to represent the tables. Eisenschlos et al., 2020 further pre-train Tapas on counterfactually-augmented and grammar-based synthetic statements.

Schlichtkrull et al., 2021 study table-based fact verification in an open-domain setting, and combine a TF-IDF retrieval model with a RoBERTa-based joint reranking-and-verification model.

Tapex (Liu et al., 2021) guides the pre-trained BART model to mimic an `SQL` executor via an execution-centric table pre-training approach. The pre-training corpus of Tapex is synthesized via sampling `SQL` queries from the SQUALL dataset (Shi et al., 2020).

SaMoE (Zhou et al., 2022) develops a mixture-of-experts network based on the RoBERTa-large model (Liu et al., 2019). The MoE network consists of different experts, and then a management module decides the contribution of each expert network to the verification result.

Volta (Gautam et al., 2021) analyzes how transfer learning and standardizing tables to contain a single header row can boost the effectiveness of table-based fact verification.

LKA (Zhao and Yang, 2022) studies the sentence-table's evidence correlation. It develops a dual-view alignment module based on the statement and table views to identify the most important words through various interactions.

4.3 Implementation Details

Our model is implemented based on the transformer architecture (Wolf et al., 2020). Specifically, we start pre-training with the public DebertaV3-Large checkpoint[5] and optimize the learning objective with Adam (Kingma and Ba, 2015). Our pre-training process runs up to 400K steps with a batch size of 16 and a learning rate of 1×10^{-6}. The complete pre-training procedure takes about 3

[5]`https://huggingface.co/microsoft/deberta-v3-large`

days on 2 RTX A6000 GPUs. For fine-tuning, the model runs up to 300K steps with a batch size of 8 and a learning rate of 5×10^{-6}.

5 Experiments and Results

5.1 Overall Performance

Table 2 summarizes the overall experimental results for various fact verification methods on TabFact. We can see that PASTA achieves the **new SOTA results** on all splits of TabFact. In particular, on the complex set containing multiple operations, PASTA largely outperforms the previous state-of-the-art by **4.7** points (**85.6% vs. 80.9%**).

Table 2 also reports the good performance of DeBERTaV3 on the table-based fact verification task. This result is analogous to the observation in Schlichtkrull et al., 2021 that RoBERTa can yield strong performance exceeding the previous closed-setting (77.6% vs. 74.4%). Both results illustrate that the BERT-like model pre-trained on textual data can also perform well on linearized tabular data. However, PASTA surpasses DeBERTaV3 by 3.1 points on the test set (89.3% vs. 86.2%), i.e., 3.9 points and 2.7 points on the simple test set and complex test set respectively. The experimental result shows that PASTA endows DeBERTaV3 with more powerful statement-table reasoning ability, which is very crucial for fact verification.

Table 3 shows that PASTA outperforms all baseline models by large margins on the SEM-TAB-FACTS dataset. In particular, PASTA significantly surpasses the DeBERTaV3 model by 5.2 points (84.1% vs 78.9%) on the test set. This shows that although our pre-training corpus only contains tables from Wikipedia, it can be applied to other domains, such as tables from scientific articles included in the SEM-TAB-FACTS dataset.

5.2 Impact of Operation Aware Pre-training

Performance on cloze pre-training. As described in Section 3.1, PASTA is pre-trained on table-operations aware cloze tasks to be capable of reasoning about table-aware operations over tables. To explore whether the model has learned such ability, we generate six test sets corresponding to different operation types, and evaluate the performance of PASTA in various steps during pre-training. The experimental results are reported in Figure 4. Overall, after 400K steps, PASTA can correctly complete more than 60% sentence-table cloze questions with various types. More specifically, with increasing

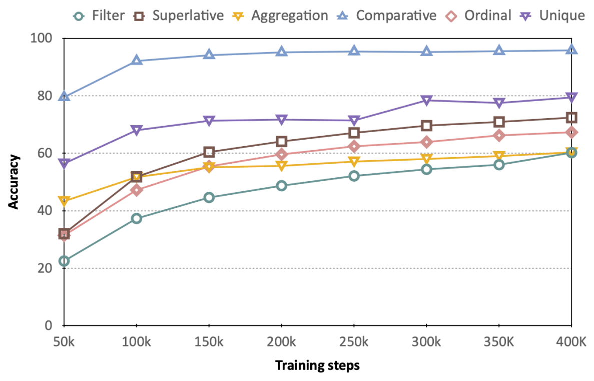


Figure 4: Accuracy on operation-aware cloze tasks at different training steps. For each operation, the size of its test set is 1K where the test set does not contain any tables from the training set.

the steps, PASTA is firstly capable of reasoning about `Comparative` operations, and finally mastering `Aggregation` and `Filter` operations. This may be attributed to the difficulty of the operations (e.g., `Aggregation` is harder than other types) and the length of token span that needs to be predicted (As shown in Table 1, `Filter` needs to predict more tokens than other types).

Operation understanding on fact verification. We further analyze whether the model can utilize the reasoning ability learned from pre-training for our downstream task, i.e., table-based fact verification. To this end, we evaluate PASTA on test sets of different operation types. We split the test set of TabFact according to the *trigger words* in the statement, which are defined in Appendix B, e.g., "*highest*" and "*lowest*" related to the `Superlative` type. We control the size of each test set as 200, while ensuring that these sets have no overlap. We compare PASTA and DeBERTaV3 on these test sets, and the results are shown in the Table 4. We can see that PASTA outperforms DeBERTaV3 on every test set, especially on the `Aggregation` type. Note that we did not use any fine-tuning strategy on both models, and thus all the improvements of PASTA are to be attributed to our table-operations aware pre-training strategy.

Comparison with Masked Language Modeling. We compare PASTA with the random masking scheme in Masked Language Modeling (MLM). For MLM, we randomly mask 15% of the tokens in a sentence-table pair, of which 10% of the masked tokens remain unchanged, 10% are replaced with randomly picked tokens, and the remainders are replaced with the `[MASK]` token. For PASTA, we

Model	Val	Test	Simple Test	Complex Test	Small Test
Table-BERT (Chen et al., 2020a)	66.1	65.1	79.1	58.2	68.1
LogicFactChecker (Zhong et al., 2020)	71.8	71.7	85.4	65.1	74.3
SAT (Zhang et al., 2020)	73.3	73.2	85.5	67.2	-
ProgVGAT (Yang et al., 2020)	74.9	74.4	88.3	67.6	76.2
Schlichtkrull et al., 2021 (Oracle retrieval)	78.2	77.6	88.9	72.1	79.4
Tapas (Eisenschlos et al., 2020)	81.0	81.0	92.3	75.6	83.9
Tapex (Liu et al., 2021)	84.6	84.2	93.9	79.6	85.9
SaMoE (Zhou et al., 2022)	84.2	85.1	93.6	80.9	86.7
DeBERTaV3	$86.1_{\pm 0.2}$	$86.2_{\pm 0.1}$	$92.8_{\pm 0.2}$	$82.9_{\pm 0.1}$	$86.5_{\pm 0.3}$
PASTA	**89.2**$_{\pm 0.4}$	**89.3**$_{\pm 0.3}$	**96.7**$_{\pm 0.2}$	**85.6**$_{\pm 0.3}$	**90.6**$_{\pm 0.2}$
Human Performance	-	-	-	-	92.1

Table 2: Performance on TabFact in terms of binary classification accuracy (%). The human performance on a small set is from Chen et al., 2020a. The notation "-" indicates that the corresponding values are not listed in the original paper. In addition, models are evaluated with 5 random runs.

Model	Val	Test
Volta (Gautam et al., 2021)	74.35	73.87
Tapas (Müller et al., 2021)	78.33	75.33
Tapex	77.53	75.47
LKA (Zhao and Yang, 2022)	80.34	78.54
DeBERTaV3	81.85	78.92
PASTA	**84.23**	**84.10**

Table 3: Performance on SEM-TAB-FACTS in terms of micro-F1 (%). The experimental results of Tapex is from Zhao and Yang, 2022.

use the masking strategy introduced in Section 3.1, which only masks the operation-aware span in the sentence. We pre-train both MLM and PASTA on DeBERTaV3. Considering pre-training efficiency, for both MLM and PASTA, we set the training step as 140K. Table 5 shows the fine-tuned results of MLM and PASTA on the TabFact dataset. We can see that PASTA outperforms MLM by a large margin on the complex set. This improvement further proves that our table-operations aware pre-training task helps the high-order symbolic reasoning in the complex set. By also observing Table 2, we find that the impact of MLM decreases slightly on the basis of DeBERTaV3 (84.9% vs. 86.2%). This may be because the random masking scheme in MLM does not work for sentence-table joint understanding, as we have already analyzed in Section 3.1.

5.3 Impact of Select-then-Rank

To verify the effectiveness of the select-then-rank method, we conduct experiments with column-wise selection and row-wise ranking on the TabFact dataset. The results of the experiment are shown in Table 6. We can see that the row-wise ranking strategy is more effective than the column-wise selection strategy. The main reason may be that the disentangled attention mechanism in DeBERTa makes the model more sensitive to the positional information of the input. The row-wise ranking strategy can put the most relevant cells in the table closer to the sentence, and thus the model can more effectively capture the sentence-table relationship.

5.4 Error Analysis

To analyze the errors of PASTA for table-based fact verification, we analyze the sentence-table pairs that PASTA predicts incorrectly in the TabFact dataset. Specifically, we consider the size of the tables and the complexity of the operations in the statements. Table 7 presents some basic statistics about the subset of the test where PASTA makes mistakes. For comparison, we also list the basic statistics about DeBERTaV3's error set and the full test set. We have the following observations. (1) Our models may not perform well on large tables. Concretely, although PASTA reduces the impact of large tables on the fact verification task compared to DeBERTaV3 (97.5 vs. 107.4), the impact of large tables on PASTA still exists compared to the average table size in the test set (97.5 vs. 89.0). (2) The number of operations in the statement is also an important cause of errors: the proportion of statements with multiple operations in PASTA's error set (16.5%) is larger than that of the overall test set (11.3%). Thus, PASTA correctly verifies most of the statements that contain only a single operation, but it still encounters difficulty to verify statements that contain multiple types of operations.

Operation	Example in TabFact	DeBERTaV3	PASTA
`Filter`	the blues and penguins game on march 20 , score was 2 - 4	88.0	**90.7** (+2.7)
`Superlative`	pacific national has the highest number in class	85.9	**86.7** (+0.8)
`Aggregation`	the average amount of points among all teams is 29	81.0	**84.5** (+3.5)
`Comparative`	ian woosnam placed higher than craig parry	85.2	**86.2** (+1.0)
`Ordinal`	the second largest number of runs was 8529	83.8	**86.9** (+3.1)
`Unique`	there are 5 different nations in the tournament	74.2	**79.1** (+4.9)

Table 4: Binary classification accuracy (%) on sentence-table pairs containing different types of operations. The six sets are sampled from TabFact based on trigger words, and each set contains 200 sentence-table pairs.

Scheme	Val	Test	Simple	Complex
MLM	$84.8_{\pm 0.2}$	$84.9_{\pm 0.2}$	$92.5_{\pm 0.1}$	$81.2_{\pm 0.2}$
PASTA	$\mathbf{87.0}_{\pm 0.1}$	$\mathbf{87.9}_{\pm 0.2}$	$\mathbf{94.0}_{\pm 0.2}$	$\mathbf{84.9}_{\pm 0.2}$

Table 5: Ablation study for the masking scheme. MLM uses random masking and PASTA uses table-operations aware masking. To avoid co-effects, we didn't use any data pre-processing method on MLM or PASTA. Models are evaluated with 5 random runs.

Method	Val	Test	Simple	Complex
PASTA	89.2	89.3	96.7	85.6
w/o col	88.9	89.0	95.8	85.5
w/o row	88.2	88.4	95.1	84.7

Table 6: Ablation study for the select-then-rank strategy. "w/o col" means that the column-wise selection strategy is not used, and "w/o row" means that the row-wise ranking strategy is not used.

	# Row	# Col	# Cell	% Mul-Ops
All Test	6.2	14.3	89.0	11.3
DeBERTaV3	6.4	16.8	107.4	13.4
PASTA	6.4	15.5	97.5	16.5

Table 7: Statistics of data collected from PASTA and DeBERTaV3's error sets. Proportion of statements that contain more than two operations is marked as "% Mul-Ops". "# Row", "# Col", "# Cell" respectively represent the average numbers of rows, columns, and cells.

6 Related Work

Sentence-Table Joint Understanding Many tasks, such as table question answering, table search, table-to-text, and table-based fact verification, require understanding tables and an `NL` sentence jointly. TAPAS (Herzig et al., 2020) and FORTAP (Cheng et al., 2021) design sentence-table joint pre-training tasks. TAPAS further leverages Whole Word Masking (WWM) to learn better representations of tables, while FORTAP leverages Numerical Reference Prediction (NRP) and Numerical Calculation Prediction (NCP) on a large corpus of spreadsheet formulas. These methods are closest to ours, but one focuses on the contextual information of `NL` and the other on the statistical characteristics of tables. Our method considers both, and improves the joint understanding of the model by means of operation-aware cloze tasks.

(Table-based) Fact Verification Program-driven methods such as LogicFactChecker (Zhong et al., 2020) and ProgVGAT (Yang et al., 2020) mainly focus on explicitly capturing logical operations in statements and representing them using a graph neural network to support fact verification. PLM-driven methods such as Table-BERT (Chen et al., 2020a), TAPAS (Eisenschlos et al., 2020), and SaMoE (Zhou et al., 2022) perform table-based fact verification tasks as an NLI task and apply a BERT-like model to encode sentence-table pairs. Our approach is more similar to SaMoE (Zhou et al., 2022). The main difference is that SaMoE uses different experts to solve different types of operations, while our method assumes that the operation combinations in statements are complex and diverse. We directly inject the atomic operations into the model through operation-aware pre-training, and then fine-tune the model to learn the various operation combinations on downstream datasets. However, PASTA is compatible with the mixture-of-experts framework, and thus we would evaluate PASTA with MoE structure on table-based fact verification datasets in the future.

7 Conclusion and Future Work

We introduced PASTA, a table-operations aware pre-training approach to train LMs for better performing fact verification over tables. PASTA achieved new SOTA results on two widely-adopted table-based fact verification benchmark datasets, TabFact and SEM-TAB-FACTS. Future work should explore how to address the challenges of more complex operations and large tables in fact verification.

Limitations

The first limitation of our work is that our synthetic pre-training corpus may lack diversity. As explained in Section 3.2, to ensure the correctness and controllability of these sentences, we generate the pre-training corpus using human-designed natural language templates. While our insight is that only atomic operations need to be learned at pre-training, which reduces the need for diversity, generating high-quality sentences with both diversity and controllability to support self-supervised learning is still a direction worth exploring.

The second limitation of our work is that fact verification is only supported on a single table. Although the TabFact (Chen et al., 2020a) dataset we used assumes that each statement can be verified by a single table, a more realistic scenario would be to combine information from multiple tables. Exploring how to do this effectively and to overcome the limitation of the input length of BERT-like models is a important direction for future work.

Ethics Statement

Dataset Collection For the pre-training dataset, we use the publicly available WikiTables (Bhagavatula et al., 2013) dataset as the table source and select high-quality relational tables from it. Then we use these tables to generate entailed statements, instead of collecting statements from the web. For the fine-tuning dataset, we use the publicly available datasets, TabFact (Chen et al., 2020a) and SEM-TAB-FACTS (Wang et al., 2021b).

Intended Use and Misuse Potential The goal of the fact verification task is to help identify misinformation. Our work focuses on verifying the statements based on analysis over tables and aims to pre-train language models to be aware of common table operations, such as aggregation over a column or comparing two tuples. It should be noted that while we treat the tables from TabFact and SEM-TAB-FACTS as trustworthy sources of evidence in the experiments, we do not assume that all of the tables in the network are trustworthy and unbiased. So, this work could also be misused through fact verification on unreliable or socially biased tables.

Broader Impact First, our operation-aware pre-training approach potentially has a broad impact on sentence-table joint understanding tasks. Second, we also notice that training a large-scale pre-trained language model requires the use of GPUs/TPUs for training, which contributes to global warming. However, we're pre-training from a public checkpoint, not from scratch. And our pre-training dataset contains only 1.2 million examples, which is very small compared to other related work (Eisenschlos et al., 2020; Liu et al., 2021).

Acknowledgements

This work was partly supported by the NSF of China (62122090, 62072461, and U1911203) and the fund for building world-class universities (disciplines) of Renmin University of China.

References

Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. 2013. Methods for exploring and mining tables on wikipedia. In *Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics, IDEA@KDD 2013, Chicago, Illinois, USA, August 11, 2013*, pages 18–26. ACM.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020a. Tabfact: A large-scale dataset for table-based fact verification. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Zhiyu Chen, Mohamed Trabelsi, Jeff Heflin, Yinan Xu, and Brian D. Davison. 2020b. Table search using a deep contextualized language model. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 589–598. ACM.

Zhoujun Cheng, Haoyu Dong, Fan Cheng, Ran Jia, Pengfei Wu, Shi Han, and Dongmei Zhang. 2021. FORTAP: using formulae for numerical-reasoning-aware table pretraining. *CoRR*, abs/2109.07323.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Julian Martin Eisenschlos, Syrine Krichene, and Thomas Müller. 2020. Understanding tables with intermediate pre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 281–296. Association for Computational Linguistics.

Devansh Gautam, Kshitij Gupta, and Manish Shrivastava. 2021. Volta at semeval-2021 task 9: Statement verification and evidence finding with tables using TAPAS and transfer learning. In *Proceedings of the 15th International Workshop on Semantic Evaluation, SemEval@ACL/IJCNLP 2021, Virtual Event / Bangkok, Thailand, August 5-6, 2021*, pages 1262–1270. Association for Computational Linguistics.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021a. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *CoRR*, abs/2111.09543.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021b. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. 2020. Tapas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4320–4333. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Qian Liu, Bei Chen, Jiaqi Guo, Zeqi Lin, and Jian-Guang Lou. 2021. TAPEX: table pre-training via learning a neural SQL executor. *CoRR*, abs/2107.07653.

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. *CoRR*, abs/1907.11692.

Thomas Müller, Julian Eisenschlos, and Syrine Krichene. 2021. TAPAS at semeval-2021 task 9: Reasoning over tables with intermediate pre-training. In *Proceedings of the 15th International Workshop on Semantic Evaluation, SemEval@ACL/IJCNLP 2021, Virtual Event / Bangkok, Thailand, August 5-6, 2021*, pages 423–430. Association for Computational Linguistics.

Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 309–319. The Association for Computer Linguistics.

Kashyap Popat, Subhabrata Mukherjee, Jannik Strötgen, and Gerhard Weikum. 2017. Where the truth lies: Explaining the credibility of emerging claims on the web and social media. In *WWW*, pages 1003–1012. ACM.

Michael Sejr Schlichtkrull, Vladimir Karpukhin, Barlas Oguz, Mike Lewis, Wen-tau Yih, and Sebastian Riedel. 2021. Joint verification and reranking for open fact checking over tables. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 6787–6799. Association for Computational Linguistics.

Shaden Shaar, Nikolay Babulkov, Giovanni Da San Martino, and Preslav Nakov. 2020. That is a known lie: Detecting previously fact-checked claims. In *ACL*, pages 3607–3618. Association for Computational Linguistics.

Tianze Shi, Chen Zhao, Jordan L. Boyd-Graber, Hal Daumé III, and Lillian Lee. 2020. On the potential of lexico-logical alignments for semantic parsing to SQL queries. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 1849–1864. Association for Computational Linguistics.

Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *SIGKDD Explor.*, 19(1):22–36.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and verification. In *NAACL-HLT*, pages 809–819. Association for Computational Linguistics.

Fei Wang, Kexuan Sun, Jay Pujara, Pedro A. Szekely, and Muhao Chen. 2021a. Table-based fact verification with salience-aware learning. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 4025–4036. Association for Computational Linguistics.

Nancy Xin Ru Wang, Diwakar Mahajan, Marina Danilevsky, and Sara Rosenthal. 2021b. Semeval-2021 task 9: Fact verification and evidence finding for tabular data in scientific documents (SEM-TAB-FACTS). In *Proceedings of the 15th Inter-*

national Workshop on Semantic Evaluation, SemEval@ACL/IJCNLP 2021, Virtual Event / Bangkok, Thailand, August 5-6, 2021, pages 317–326. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Xiaoyu Yang, Feng Nie, Yufei Feng, Quan Liu, Zhigang Chen, and Xiaodan Zhu. 2020. Program enhanced fact verification with verbalization and graph attention network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7810–7825. Association for Computational Linguistics.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. Tabert: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8413–8426. Association for Computational Linguistics.

Seunghyun Yoon, Kunwoo Park, Joongbo Shin, Hongjun Lim, Seungpil Won, Meeyoung Cha, and Kyomin Jung. 2019. Detecting incongruity between news headline and body text via a deep hierarchical encoder. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 791–800. AAAI Press.

Hongzhi Zhang, Yingyao Wang, Sirui Wang, Xuezhi Cao, Fuzheng Zhang, and Zhongyuan Wang. 2020. Table fact verification with structure-aware transformer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1624–1629. Association for Computational Linguistics.

Guangzhen Zhao and Peng Yang. 2022. Table-based fact verification with self-labeled keypoint alignment. In *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*, pages 1401–1411. International Committee on Computational Linguistics.

Wanjun Zhong, Duyu Tang, Zhangyin Feng, Nan Duan, Ming Zhou, Ming Gong, Linjun Shou, Daxin Jiang, Jiahai Wang, and Jian Yin. 2020. Logicalfactchecker: Leveraging logical operations for fact checking with graph module network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6053–6065. Association for Computational Linguistics.

Yuxuan Zhou, Xien Liu, Kaiyin Zhou, and Ji Wu. 2022. Table-based fact verification with self-adaptive mixture of experts. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 139–149. Association for Computational Linguistics.

A Details of Pre-training Corpus

For the six table-aware operation types, we design a total of 50 NL-SQL template pairs to generate sentence-table instances for pre-training. Table 8 shows two examples of each type of operations. We first populate the NL-SQL template pairs with the specific content in the table T, and then the context sensitive words w' in the NL template are selected by a fixed pre-trained LM in its synonym set to improve the fluency of the generated sentences. We define four context-sensitive word candidate sets, which are presented in Table 9.

B Trigger Words Definition

Table 10 shows the trigger words we have used to identify different operation types in Section 5.2. These trigger words are expanded from the trigger words defined in TabFact (Yin et al., 2020), and are then classified according to the six table-aware operation types introduced in this paper.

Operation	NL Template	SQL Template
Filter	[Value2]'s [Column1] is [ANS].	SELECT [Column1] FROM T WHERE [Column2] = [Value1]
	the [Column1] of [Value2] is [ANS].	SELECT [Column1] FROM T WHERE [Column2] = [Value1]
Superlative	the highest [Column1] is [ANS]	SELECT MAX([Column1]) FROM T
	[ANS] has the highest [Column2] of all [Column1]	SELECT [Column1] FROM T ORDER BY [Column2] DESC LIMIT 1
Aggregation	the sum of [Column1] when [Column2] is [Value2] is [ANS].	SELECT SUM([Column1]) FROM T WHERE [Column2] = [Value2]
	the average of [Column1] when [Column2] is [Value2] is [ANS].	SELECT AVG([Column1]) FROM T WHERE [Column2] = [Value2]
Comparative	[Column1] [ANS]'s [Column2] is higher than [Value2]	SELECT [Column1] FROM t WHERE [Column2] > [Value2]
	[ANS] has higher [Column2] than [Value2]	SELECT [Column1] FROM t WHERE [Column2] > [Value2]
Ordinal	[ANS] has the second highest [Column2]	SELECT [Column1] FROM T WHERE [Column2] < (SELECT MAX([Column2]) FROM T) ORDER BY [Column2] DESC LIMIT 1
	[ANS] has the second lowest [Column2]	SELECT [Column1] FROM T WHERE [Column2] > (SELECT MIN([Column2]) FROM T) ORDER BY [Column2] ASC LIMIT 1
Unique	there are [ANS] different [Column1] on the list.	SELECT COUNT(DISTINCT [Column1]) FROM T
	the total number of different [Column1] is [ANS].	SELECT COUNT(DISTINCT [Column1]) FROM T

Table 8: Examples of NL Templates and SQL Templates for each table-aware operation type. The words in each NL Template that need to be polished by the pre-trained LM are marked in red. The operation-aware tokens that need to be predicted during pre-training are underlined.

w'	Candidate Set
"highest"	"highest", "most", "biggest", "largest", "oldest", "greatest", "heaviest", "longest", "tallest"
"lowest"	"lowest", "least", "smallest", "youngest", "shortest"
"higher"	"higher", "more", "bigger", "larger", "older"
"less"	"less", "smaller", "lower", "younger"

Table 9: Four context-sensitive word candidate sets. w' refers to the context-sensitive word in the NL templates.

Operation	Trigger Words
Filter	"is", "was", "are", "were"
Superlative	"lowest", "least", "smallest", "youngest", "shortest", "first", "best", "newest", "latest"
Aggregation	"average", "count", "sum", "total"
Comparative	"than", "higher", "more", "bigger", "larger", "older", "less", "smaller", "lower", "younger", "same", "equal"
Ordinal	"second", "third", "fourth"
Unique	"different"

Table 10: Trigger words for each table-aware operation type.