AttesTable at SemEval-2021 Task 9: Extending Statement Verification with Tables for Unknown Class, and Semantic Evidence Finding

Harshit Varma and **Aadish Jain**
Indian Institute of Technology Bombay
Mumbai, Maharashtra, India
{harshitvarma,aadishjain}@cse.iitb.ac.in

Pratik Ratadiya and **Abhishek Rathi**
vCreaTek Consulting Services Pvt Ltd
Pune, Maharashtra, India
{pratik.r,abhishek.r}@vcreatek.com

Abstract

This paper describes our approach for Task 9 of SemEval 2021: *Statement Verification and Evidence Finding with Tables*. We participated in both subtasks, namely statement verification and evidence finding. For the subtask of statement verification, we extend the TAPAS model to adapt to the 'unknown' class of statements by finetuning it on an augmented version of the task data. For the subtask of evidence finding, we finetune the DistilBERT model in a Siamese setting.

1 Introduction

Tables provide a compact and structured way of presenting information. They are easily interpretable by humans and are widely used in scientific papers, business articles, and even in government reports. At the same time, it is easy to misinterpret tabular data and even use it maliciously to spread misinformation. Thus, systems that can verify facts from tables and locate evidence in them can go a long way in ameliorating issues related to table interpretation. Such systems can also be used for question-answering over large tables, that are difficult to analyze manually, since the task of fact verification with evidence finding is closely related to that of question answering (Jobanputra, 2019).

Table entailment refers to the task of finding whether a sentence is refuted or supported by a given table. Traditionally, it has been considered a binary classification task. However, there could be instances where the table is not capable enough to either refute or accept the given statement, meaning the statement context is "unknown" for the table. In this paper, we have presented a table entailment setup that extends to three classes: refuted, entailed, and unknown, as a part of the Sem-Tab-Fact task (Wang et al., 2021). The task could be divided into two parts: statement verification, and evidence finding. Given a table and a statement, one has to first determine whether the table entails the statement, refutes the statement, or has insufficient information about it. This forms the subtask of *statement verification*. If the table entails or refutes the statement, one would also like to know which cells in the table provided evidence for the same. This is the subtask of *evidence finding*, that can be formulated as a binary classification problem. Each cell in the table is assigned one of the 'relevant' or 'irrelevant' labels depending on whether it provided evidence for the given statement.

Recent years have seen a rise in the use of transfer learning in language processing (Malte and Ratadiya, 2019a) owing to their superior performance. Models based on underlying concepts like attention mechanism and transformers are seeing widespread use across a range of tasks (Malte and Ratadiya, 2019b; Ratadiya and Mishra, 2019). Our findings concurred with this trend as we used similar kinds of architectures as the fundamental blocks in the systems for both the subtasks. For the subtask of *statement verification*, we modify the recently released TAPAS model (Eisenschlos et al., 2020) that is pre-trained on the TabFact dataset (Chen et al., 2020). The pre-trained model is trained on only two classes: 'entailed' and 'refuted', and is not capable of classifying the 'unknown' samples. The training data provided to us for our task also contains the same two labels. To adapt to the 'unknown' statements, we augment the given data by including random statements from other tables as unknown. We then fine-tune the TAPAS model by extending it to three classes on this augmented data.

Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), pages 1276–1282
Bangkok, Thailand (online), August 5–6, 2021.

Using this approach, we achieve a three-way F1 score of 65.59 and a two-way F1 score of 71.72 on the test data. We ranked 8^{th} on the official leaderboard for this subtask.

For the subtask of *evidence finding*, we finetune the DistilBERT model (Sanh et al., 2019) in a Siamese setup (Reimers and Gurevych, 2019). The model is pre-trained on the SNLI dataset (Bowman et al., 2015), the MultiNLI dataset (Williams et al., 2018) and the STS benchmark (Cer et al., 2017). For finetuning, we use the Contrastive loss (Hadsell et al., 2006) as our loss function. For making a prediction, we calculate the cosine similarity between the embeddings computed by the model for both the statement and the cell value. If the cosine similarity value crosses a set threshold, we consider the cell to be relevant to the given statement. Using this approach, we achieve an F1-score of 43.02 on the test data and secured the 7^{th} rank on the official leaderboard for this subtask.

The source code of our proposed approach for both the subtasks has been made public[1] to encourage usage and improve the reproducibility of the results.

2 Related Work

Recently, the TabFact dataset was released, which contained 118K human-annotated statements, related to about 16K Wikipedia tables. These statements were annotated for only two labels, 'entailed' and 'refuted', leaving out the 'unknown' cases. A system's ability to distinguish 'unknown' statements from 'entailed' and 'refuted' is quite critical, as one may elusively create seemingly believable statements that are actually 'unknown'.

Recently, there has been work in areas related to table fact verification, especially since the release of the TabFact dataset. Many of these approaches are graph-based in nature (Shi et al., 2020; Yang et al., 2020; Zhong et al., 2020). The current state-of-the-art on the TabFact dataset is the recently released TAPAS model, which outperforms its predecessor by approximately 6%. Unfortunately, all of these models are quite far from human-level performance, suggesting ample scope for improvement.

Little to no work has been done previously on the task of evidence finding, with the closest approaches being related to table-based question answering like WikiTableQuestions (Pasupat and Liang, 2015), WikiSQL (Zhong et al., 2018) and SQA (Iyyer et al., 2017) where getting the answer for a question can be considered analogous to getting evidence for a statement.

3 System Overview

3.1 Data and Preprocessing

The training dataset for this task contains two kinds of data:

- **Manual annotations:** These are crowdsourced from human annotators and have been validated by a second round of validation.
- **Auto-generated annotations:** These statements are auto-generated using a random paraphraser and table understanding service (Zheng et al., 2020).

A separate development set was also provided.

3.1.1 Subtask A: Statement Verification

The provided dataset for this task is in XML format. We use a custom parser to convert the data into CSV format, such that each data sample is of the form (*table, statement, label*), where the table is also a CSV, extracted from the corresponding XML file. No preprocessing is applied to the statements.

In general, the table cells can span across multiple rows and columns. To simplify things, we treat a cell spanning multiple columns/rows as multiple cells with the same value. This preserves the logical hierarchy in the table while keeping the table structure simple. See Figure 1 for an example.

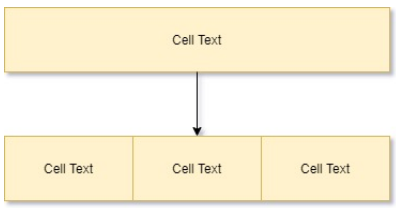

Figure 1: A cell spanning three columns is considered as three separate cells with the same value

A table may also have additional metadata accompanying it, like the legend, the caption, and the footer. Our final model does not use this metadata since we observe in Section 5.1.2 that including the metadata has a small negative impact on the model performance.

After the above preprocessing, we have 179,345 autogenerated and 4506 manual data samples.

[1]The code is available at https://www.github.com/vcreatek/attestable-semeval/

3.1.2 Subtask B: Evidence Finding

For this subtask, we prepare the data in the form (*cell text*, *statement text*, *label*), where the label can be 'relevant' or 'irrelevant'. We do not encode the table's structural information and focus only on the semantic similarity between the statements and the cell texts.

An important point to note here is that only a few of the cells in the table actually contribute to the evidence for each statement. The average ratio of the number of relevant samples to total samples as observed in the development set for each statement is around 0.097. This causes a substantial class imbalance since a meager 7% of the total samples are 'relevant'. To attend this problem, we undersample the 'irrelevant' samples by removing some irrelevant samples for each statement from the data. We try three ratios for undersampling and compare the percentage of relevant statements after undersampling in Table 1.

$\mathbf{n_i'}$	**% r**	$\mathbf{n_t}$
$0.6n_i$	10.15	10.24
$0.5(n_r + n_i)$	11.24	9.25
$0.4n_i$	14.54	7.15

Table 1: n_i and n_r stand for the number of irrelevant and relevant samples for a statement. n_i' stands for the no. of irrelevant samples kept for each statement after the undersampling. $\%r$ stands for the percentage of relevant samples after undersampling. n_t stands for the total samples after undersampling, in millions.

In the original data, the average ratio of relevant to total samples is around 9.7%, so we would want our data to reflect a similar ratio. Therefore, we use the data undersampled using the first approach for further analysis. Hence, the total number of samples after undersampling stand at 10.24M.

3.2 Data Augmentation

The given data has no 'unknown' statements; thus, we augment the data to introduce samples of this class label. Let S denote the set of all statements in our data. Let S_e denote the set of all entailed statements in our data. Let S_r denote the set of all refuted statements in our data. Then $S = S_e \cup S_r$.

For a given table t, let S_t denote the set of all statements associated with that table. We first calculate n_e (the number of entailed statements for that table) and n_r (the number of refuted statements for that table). We then calculate n_u (the number of unknown statements to add) as follows:

$$n_u = \max(\min(n_e, n_r), 1) \quad (1)$$

After this, we randomly select n_u statements from $S \backslash S_t$. This forms the set of 'unknown' statements for the table t. We do this for all tables in the data.

The above augmentation scheme ensures that the resultant augmented data has an almost equal overall distribution among the three classes. It also ensures that evenness across tables is maintained, as sufficient unknown statements (at least one) are added for each table, and not just overall. After performing the above augmentation scheme, we get $85,296$ unknown statements for the autogenerated data and $1,637$ unknown statements for the manual data.

3.3 Models

3.3.1 Subtask A: The TAPAS Model

TAPAS (Herzig et al., 2020) is a recently released BERT (Devlin et al., 2019) based model that encodes the structural information via column, row and rank embeddings. Eisenschlos et al. extended TAPAS for binary entailment using the TabFact dataset. We use the base variant (12 encoder blocks) of this TAPAS model, which has been pretrained on the TabFact dataset. We use the base variant over the large variant (24 encoder blocks) due to limitations on computation power and due to the superior performance of the base variant as shown in Section 5.1.2.

For extending the TAPAS model to support the unknown class, we replace the two-neuron linear layer in the pre-trained model by a randomly initialized linear layer consisting of three neurons as the output layer. We then finetune this modified model on the augmented data containing three classes. Figure 2 shows the overview of the modified model. It consists of an embedding layer that concatenates the positional and semantic information of a cell value, the encoder block consisting of transformer layers and subsequent layers that are required for classification output. The core layers of the modified model are in accordance with the original TAPAS model (Herzig et al., 2020; Eisenschlos et al., 2020).

3.3.2 Subtask B: Siamese DistilBERT

For the evidence finding subtask, we finetune the DistilBERT model on the undersampled data. The

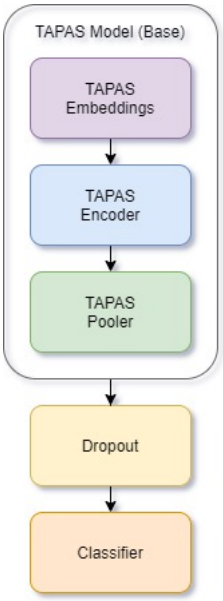

Figure 2: The modified TAPAS architecture. 'Classifier' is just a linear layer with three neurons

model was pre-trained with mean pooling on the SNLI dataset, the MultiNLI dataset, and the STS benchmark. The mean pooling is applied in a Siamese setting using the Contrastive loss as the loss function.

Let s denote a statement, let c denote the cell, let r denote the relevancy/label (this is either 0 or 1) and let $\mathcal{M}(x)$ denote the embedding computed the model for an input x. Then, the distance d and the Contrastive loss $\mathcal{L}(d, r)$ is defined as follows:

$$d = \textsc{CosineDistance}(\mathcal{M}(s), \mathcal{M}(c))$$
$$\mathcal{L}(d, r) = \frac{1}{2}\left((1-r)d^2 + r(\max(0, m-d))^2\right) \quad (2)$$

Here, m denotes the 'margin' value, which ensures that the dissimilar pairs contribute to the loss only if their distance is within this margin. For making inferences, we first use our finetuned model to compute the sentence embeddings for the statement and the cell text and then compute the cosine similarity between the two. Hyperparameter tuning and other details about the models are discussed in Section 4.

4 Experimental Setup

4.1 Hyperparameters and Data Splits

4.1.1 Subtask A

For training, we first merge the autogenerated and manual augmented data. We then perform an $80-20$ split on this merged data to get our training and validation sets. We then fine-tune on the training data and validate on the validation set.

The 'TAPAS Encoder' (Figure 2) consists of a stack of 12 encoder blocks (similar to the BERT Base). For finetuning, we freeze the entire model, except the last three encoder blocks, the 'TAPAS Pooler', and the final classification layer.

This leaves 22M trainable parameters and 89M frozen (i.e. untrainable) parameters.

Categorical Cross-Entropy is used as the loss function. We use a batch size of 32 and finetune for 3 epochs using the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 5×10^{-5}.

Before making the final submission, the training and development sets are merged together and the model is trained for one more epoch.

4.1.2 Subtask B

We take a 500K sized randomly selected sample from the undersampled data and perform an $80-20$ split on this to get our training and validation sets. We keep the margin m as 0.5 and finetune the entire model, with no parameters frozen, for a single epoch using a batch size of 64. For making the inferences from the cosine similarity, we use an empirically selected threshold of 0.3.

4.2 Libraries and Tools

Google Colab[2] was used to perform all the experiments. The time taken per epoch for any model did not exceed 10 hours. The GPUs automatically allotted by Colab kept varying between Tesla T4, Tesla P100-PCIE-16GB, and Tesla K80. PyTorch[3] is used as the central framework for both the tasks. For subtask A, Huggingface's Transformers library[4] was used to load the pre-trained TAPAS model. For subtask B, we use the SentenceTransformers framework[5] to load the pretrained Siamese DistilBERT model.

5 Results

5.1 Subtask A

5.1.1 Official Metrics

The original pre-trained TAPAS (Base) model, without any finetuning, achieves a two-way F1 score of 62.56. Our final finetuned model achieves

[2]Free version

[3]Version 1.7.0: https://pytorch.org/docs/1.7.0/

[4]Version 4.1.1: https://huggingface.co/transformers/v4.1.1/

[5]Version 0.4.1: https://www.sbert.net/

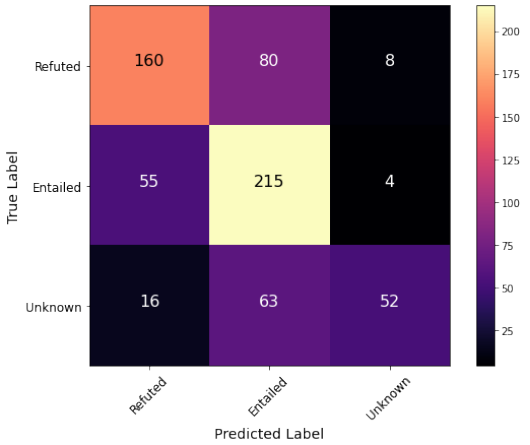

Figure 3: Confusion matrix for the submitted model

a two-way F1 score of 71.72 and a three-way F1-score of 65.59. Figure 3 shows the corresponding confusion matrix. We ranked 8^{th} on the official leaderboard for subtask-A.

From the confusion matrix, we observe that the most confusing cases for our model are true 'unknown' statements, which get classified as 'entailed'. On manual analysis of such statements, we observe that many of these have a considerable textual overlap with the table. This misclassification seems to be a consequence of our augmentation scheme. The statements that we add as 'unknown' while augmentation have almost no textual overlap with the table data since they were sourced from other tables. Although, in general, as observed in the test data, unknown statements can broadly be classified into two kinds: 'related' and 'unrelated'. The former being unknown statements that are related in a semantic or a directly textual way to the table's data, and the latter are not related to the table's data in any way. The following example should make this clear.

Orders	Week 1	Week 2	Week 3
Order 1	Peat	Straw	Silage
Order 2	Straw	Silage	Control
Order 3	Silage	Control	Combo

Table 2: A table containing data about a few weekly orders

In Table 2, a 'related' unknown statement will be "A total of 10L of **silage**, **straw**, **peat** or **combo** was distributed". The boldfaced words overlap directly with the table's contents
While, an 'unrelated' unknown statement can be "Earth revolves around the Sun". Thus, both of these sub-classes of the 'unknown' class are important. Synthesizing 'unrelated' unknown statements is a simple task, whereas synthesizing 'related' statements without any additional information seems to be a fairly non-trivial task.

We also observe that wrongly classified statements are, on average, 8.8 characters longer than correctly classified statements. Figure 4 shows this histogram.

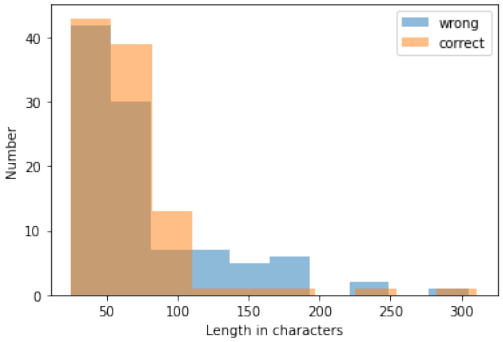

Figure 4: Histogram of lengths (in characters) of 100 randomly selected correctly classified and wrongly classified statements

5.1.2 Ablation Study

Effect of Table Metadata

A table may have additional metadata like the caption, legend, and footer associated with it. For simplicity, we include it directly inside the table as rows. If the table has a caption, we add it as a single row at the top of the table. Similarly, if it has a legend, it goes as a single row below the caption, and the footer is added as a row at the end of the table. Keeping other hyperparameters the same, we get the following results:

Included Metadata	Validation Accuracy
Yes	0.92
No	0.93

Table 3: Impact of inclusion of metadata on validation accuracy

We observe that including the metadata worsens the accuracy. A possible reason could be our way of including the metadata as rows. There may be other better ways of doing this, which may further improve the accuracy.

Effect of Model Size

We try out two variants of TAPAS: Base and Large. The Base variant has 12 encoder blocks, and the

Large variant has 24, similar to BERT Base and BERT Large. On the TabFact dataset, the Base variant is only about 0.24 points behind the Large one. The following tests are performed on the original, unaugmented data, containing two classes[6]. We merge the original manual and autogenerated statements into a single dataset and perform an $80-20$ split.

Variant	Validation Accuracy
Base	0.81
Large	0.71

Table 4: Effect of finetuning a larger model

5.2 Subtask B

Our final model achieves an F1 score of 43.02. Figure 5 shows the corresponding confusion matrix. As evident from the confusion matrix, there is a lot of room for improvement, especially in handling the 'relevant' statements. We ranked 7^{th} on the official leaderboard for subtask-B.

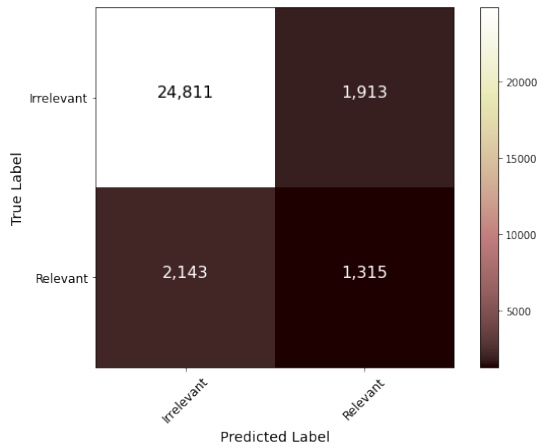

Figure 5: Confusion matrix for Subtask B

6 Conclusion

Thus we have presented a statement verification and evidence finding setup for tables. For subtask-A, we extended the TAPAS model to adapt to the 'unknown' statements. For subtask-B, we used a semantic approach for evidence finding. Our results for subtask-A show the problems encountered while generating and working with the 'unknown' statements. For subtask-B, our results show the effect of taking only the semantic information into account. An important future prospect for subtask-A would be to find a more effective way of generating the 'unknown' statements. For subtask-B, utilizing the table's available structural information to improve the results seems to be a promising prospect.

[6]Training the Large variant on augmented data exceeds the time limit of Google Colab (12 hours)

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020. Tabfact: A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations*.

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.

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

R. Hadsell, S. Chopra, and Y. LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742.

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.

Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang. 2017. Search-based neural structured learning for sequential question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1821–1831, Vancouver, Canada. Association for Computational Linguistics.

Mayank Jobanputra. 2019. Unsupervised question answering for fact-checking. In *Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER)*, pages 52–56, Hong Kong, China. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Aditya Malte and Pratik Ratadiya. 2019a. Evolution of transfer learning in natural language processing. *arXiv preprint arXiv:1910.07370*.

Aditya Malte and Pratik Ratadiya. 2019b. Multilingual cyber abuse detection using advanced transformer architecture. In *TENCON 2019-2019 IEEE Region 10 Conference (TENCON)*, pages 784–789. IEEE.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480, Beijing, China. Association for Computational Linguistics.

Pratik Ratadiya and Deepak Mishra. 2019. An attention ensemble based approach for multilabel profanity detection. In *2019 International Conference on Data Mining Workshops (ICDMW)*, pages 544–550. IEEE.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Qi Shi, Yu Zhang, Qingyu Yin, and Ting Liu. 2020. Learn to combine linguistic and symbolic information for table-based fact verification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5335–5346, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Nancy Xin Ru Wang, Diwakar Mahajan, Marina Danilevsky, and Sara Rosenthal. 2021. Semeval-2021 task 9: Fact verification and evidence finding for tabular data in scientific documents (sem-tab-facts). In *Proceedings of the 15th international workshop on semantic evaluation (SemEval-2021)*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. 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)*, pages 7810–7825, Online. Association for Computational Linguistics.

Xinyi Zheng, Doug Burdick, Lucian Popa, Xu Zhong, and Nancy Xin Ru Wang. 2020. Global table extractor (gte): A framework for joint table identification and cell structure recognition using visual context.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Seq2SQL: Generating structured queries from natural language using reinforcement learning.

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*, pages 6053–6065, Online. Association for Computational Linguistics.