2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

Cross-Document Fact Verification Based on Fine-Grained Graph Neural Network

Xiaoman Xu, Xiaoxu Zhu and Peifeng Li

School of Computer Science and Technology, Soochow University, Suzhou, China
20224227077@stu.suda.edu.cn, {xiaoxzhu, pfli}@suda.edu.cn

Abstract. Cross-Document Fact Verification (CDFV) aims to retrieve evidence from multiple documents to verify the factuality of a given claim. However, existing CDFV approaches fail to capture complex semantic relationships and fine-grained information in the evidence. To address these issues, we propose a Fine-Grained Graph Neural Network (FGGNN) for CDFV. FGGNN constructs a sentence-level graph during the evidence selection stage and efficiently propagates information within the graph using Graph Attention Networks (GAT), accurately capturing the complex relationships between sentences. This enables FGGNN to select trustworthy and relevant evidence. In the claim verification stage, FGGNN constructs a word-level evidence graph to capture fine-grained relationships at the word level. It then uses a Relational Graph Convolutional Network (RGCN) [1] to propagate and update information within the graph, fully uncovering the potential logic in the evidence. Additionally, an attention mechanism is introduced to weight the evidence based on its relevance to the claim, emphasizing the importance of key evidence. Finally, FGGNN considers all the evidence and claim information to accurately predict the label of the claim. Experimental results on the CHEF dataset demonstrate the effectiveness of FGGNN in achieving accurate fact verification.

Keywords: Fact Verification, Evidence selection, Claim Verification.

1 Introduction

The exponential growth of online information has raised significant concerns about the proliferation of false and misleading content. Claim Verification with Document-based Fact-checking (CDFV) addresses this challenge by automatically retrieving evidence from public sources to validate claims [2]. A typical fact verification system categorizes claims into three labels: Supported, Refuted, or Not Enough Information, based on whether the evidence confirms, contradicts, or is insufficient to judge the claim.

In the CHEF corpus [3], gold-standard evidence comprises manually annotated sentences from documents, serving as ground truth for validating claims. Recent advances in fact verification have primarily adopted a three-stage pipeline: document retrieval, evidence selection, and claim verification [4]. Prior research [5-10] has focused on fine-tuning pre-trained language models (PLMs) or using homogeneous graph-based models. In the fine-tuning approach, PLMs are fine-tuned with concatenated evidence. In the graph-based approach, a fully connected evidence graph is constructed [11-13], where nodes represent individual evidence, and a graph neural network (GNN) propagates neighborhood information to aggregate semantic representations.

Although these existing methods have shown certain effectiveness, they suffer from two major drawbacks. Firstly, fact verification requires the capture of semantic relationships among diverse evidence items, which in turn demands sophisticated modeling of the connections between evidence. Transformer-based approaches often prove inadequate as they simply concatenate evidence or handle claim-evidence pairs in isolation, failing to thoroughly explore the complex interconnections between different pieces of evidence. Secondly, most graph-based methods construct sentence-level graphs with claim-evidence pairs as nodes and use PLMs to represent these nodes [14]. These models have limitations in capturing fine-grained details within the evidence. They primarily function at the sentence level, overlooking the rich semantic information hidden at the word and phrase levels.

To deal with the previously mentioned challenges, we propose a novel Fine-Grained Graph Neural Network (FGGNN) for fact verification. Specifically, for documents, we decompose them into sentences and construct a sentence-level graph. We use GNN to conduct an in-depth exploration of the relationships between sentences, thereby filtering out valid evidence. This approach fully considers the contextual information of sentences within the documents and their potential semantic connections, enabling it to locate key evidence more accurately than traditional methods. For the evidence, to capture fine-grained relationships between words, we construct a word-level evidence graph. We use RGCN [1] to propagate and update information, thereby obtaining a more comprehensive and accurate semantic representation of the evidence. Then, we employ an attention mechanism to allocate weights to diverse evidence segments., highlighting the role of key evidence. Finally, we fuse the weighted evidence representation with the claim representation, comprehensively considering all relevant information to classify the claim's label. The primary contributions of our research are presented below:

1. We create an evidence selection component that can efficiently choose evidence sentences from multiple documents. It captures sentence semantic relationships and learns from gold evidence to boost selection accuracy.

2. We propose a claim verification module using word-level evidence graph and RGCN [1] to capture fine-grained relationships and integrate evidence for verification.

3. Thorough experiments carried out on established benchmark datasets reveal that our method outperforms the current state-of-the-art approaches, demonstrating its effectiveness and superiority in CDFV.

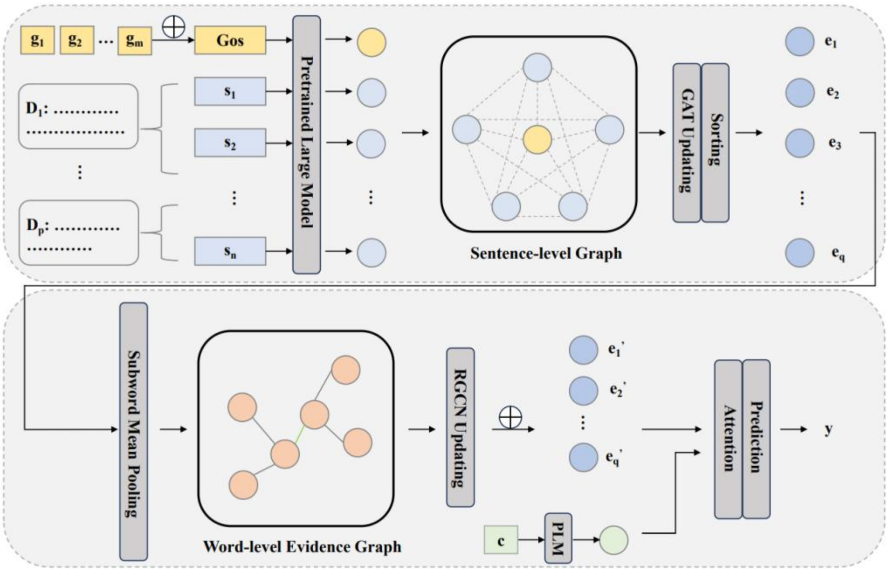

Fig. 1. Architecture of FGGNN.

2 Method

This section outlines the framework of FGGNN through three components: task definition, evidence retrieval, and claim verification. The architectural overview is illustrated in Figure 1.

2.1 Task Definition

Fact verification requires determining the authenticity of a claim (c) by analyzing evidence extracted from document collection $\mathrm{D}=\{\mathrm{d}_1, \mathrm{d}_2, \cdots, \mathrm{d}_\mathrm{p}\}$ and predefined gold evidence set $\mathrm{Gold}=\{\mathrm{g}_1, \mathrm{g}_2, \cdots, \mathrm{g}_\mathrm{m}\}$. The process involves two stages: 1) Extracting relevant evidence sentences $\mathrm{E}=\{\mathrm{e}_1, \mathrm{e}_2, \cdots, \mathrm{e}_\mathrm{q}\}$ from documents, and 2) Assigning a truthfulness label y∈{S(support), R(refute), N(not enough information)} based on E.

2.2 Evidence Selection

Sentence Encoder. For each claim, to integrate information from different passages, we decompose each document into sentences and compile them into a sentence set $\mathrm{S} = \{\mathrm{s}_1, \mathrm{s}_2, \cdots, \mathrm{s}_\mathrm{n}\}$. We utilize the BERT encoder to generate semantic representations for each sentence in both the claim and the sentence set.

For a given claim c and its associated sentence set S, as well as the gold evidence set $Gold$, we generate sentence embeddings by encoding each sentence independently using BERT. Specifically, each sentence $s_i \epsilon S$ and $g_i \epsilon Gold$ is prepended with $[CLS]$ and appended with $[SEP]$, using BERT positional embeddings for contextual representations.

$$\mathrm{h_s^i = BERT([CLS]\ s_i\ [SEP])} \tag{1}$$

where i ranges from 1 to the number of sentences in S.

$$\mathrm{h_g^j = BERT([CLS]\ g_j\ [SEP])} \tag{2}$$

To obtain the overall representation of the gold evidence sentence, we apply a pooling operation to the output of the token embedding by BERT, resulting in the sentence embedding of $Gold$:

$$\mathrm{h_g^{pooled}} = \frac{1}{\mathrm{m}} \sum_{\mathrm{j=1}}^{\mathrm{m}} \mathrm{h_j} \tag{3}$$

where j ranges from 1 to the number of sentences in $Gold$.

Graph Construction. Due to the large amount of textual information, each sentence becomes a graph node, with gold evidence sentences labeled 1 (non-selectable super-node) and others labeled 0. To learn the information from gold evidence, we treat the gold evidence set as a single node $\mathrm{v_g}$ in the graph. We set its label as 1, but it will not be selected as evidence during the evidence output stage. For node i: we have $v_i = (x_i, y_i)$, where $\mathrm{x_i}$ serves as the embedding representation of the node and y_i is its corresponding label. An edge is established between every pair of nodes. The set of edges in the graph is denoted as $E = \{(v_i, v_j) | v_i \in V, v_j \in V, i \neq j\}$.

Information Propagation. In order to extract information from diverse types of nodes, we employ GAT to capture distinct node relations. For each node v_i, we calculate the attention weight α_{ij} between it and its neighbor node v_j using the following formula:

$$\mathrm{\alpha_{ij}} = \frac{\mathrm{exp(LeakyReLU(a^T\ [W\ h_i||\ W\ h_j\ ])}}{\sum_{\mathrm{k\in N(i)}} \mathrm{exp(LeakyReLU(a^T\ [W\ h_i \parallel W\ h_k\ ])}} \tag{4}$$

where W ϵ $\mathbb{R}^{F' \times F}$ denotes a linear transformation matrix that projects the original node features h_i of the node into a new feature space F'. ‖ represents the concatenation operation. a ϵ $\mathbb{R}^{2F'}$ is a learnable vector. $\mathcal{N}(i)$ represents the set of neighboring nodes of v_i.

Following the computation of attention weights α_{ij}, the representation of node v_i is updated through an aggregation of its neighboring nodes' features:

$$h_i' = \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij} W\, h_j\right) \tag{5}$$

where h_i' denotes the refined feature representation of node v_i. σ is an activation function.

The multi-head attention mechanism is adopted. Specifically, for each attention head k, independent attention weights and corresponding node representation updates are computed. Afterwards, these outputs are combined via concatenation:

$$\mathrm{h_i^{'}} = ||_{\mathrm{k=1}}^{\mathrm{K}} \sigma\left(\sum_{\mathrm{j\in N(i)}} \alpha_{\mathrm{ij}}^{\mathrm{(k)}} \mathrm{W^{(k)}\ h_j}\right) \tag{6}$$

where K represents the number of multi-head attentions.

This concatenated representation will be fed into the next GAT layer. Upon reaching the final layer of the model, we can acquire the ultimate representation of node i, which is denoted as h_i^L.

Veracity Prediction and Evidence Output. The prediction formula is as follows:

$$\hat{y}_i = \sigma\left(W_{out} h_i^L\right) \tag{7}$$

where W_{out} serves as the weight matrix of the classification layer, while $\hat{y}_i$ is the predicted value of node v_i.

The loss function is:

$$\mathcal{L} = -(y_i log(\hat{y}_i) - (1 - y_i) log(1 - \hat{y}_i)) \tag{8}$$

where y_i represents the actual label, while $\hat{y}_i$ denotes the probabilistic prediction for node v_i.

The graph nodes are ranked by their probability of being labeled as 1, and the top q sentences are selected as the final output evidence.

2.3 Claim Verification

Word-level Evidence Graph Construction. To obtain a more granular understanding of the evidence, we focus on capturing fine-grained relationships at the word level. This approach enhances the effectiveness of claim verification. We first tokenize the evidence and treat each word as a node in the graph. A word-level graph is constructed where each word serves as a node. The node set of the graph is denoted as $V=\{w_1, w_2, \cdots, w_m\}$. The connections between words are carefully defined. For adjacent words in the same sentence, syntactically and semantically an edge is created between them: $E_{intra} = \{(w_i, w_{i+1}) \mid w_i, w_{i+1} \epsilon\, s_j\}$. Additionally, if the same word appears in different sentences, to emphasize semantic coherence and cross-sentence relationships, an edge is also constructed: $E_{inter} = \{(w_i, w_k) \mid w_i \epsilon\, s_j, w_k \epsilon\, s_l, w_i = w_k, j \neq l\}$. This graph structure effectively captures both the internal relationships between words within sentences and the semantic links across sentences, providing a solid foundation for subsequent information integration.

Information Propagation. We adopt the RGCN [1] framework for node representation refinement. The iterative update rule for node embeddings is formalized as:

$$h_i^{(l+1)} = \sigma\left(\sum_{r \epsilon R} \sum_{j \epsilon N_r(i)} \frac{1}{c_{i,r}} W_r^{(l)} h_j^{(l)} + W_0^{(l)} h_i^{(l)}\right) \tag{9}$$

where $h_i^{(l)}$ denotes the embedding of node i at layer l. R represents the set of edge types in the heterogeneous graph. $W_r^{(l)}$ is the trainable weight matrix specific to edge type r, $W_0^{(l)}$ is the self-loop weight matrix capturing node-specific features. $c_{i,r}$ is a normalization constant to balance contributions from different neighbors.

Evidence Readout. After the representations of word nodes are updated, we concatenate the word nodes within the same piece of evidence to generate a new representation for that evidence. Suppose an evidence sentence e_i contains word nodes

$w_1, w_2, \cdots, w_m$. Then the representation h_{e_i} of this evidence can be obtained by concatenating the representations of each node:

$$\mathrm{h}_{\mathrm{e}_\mathrm{i}} = \mathrm{Concat}\left(\mathrm{h}_{\mathrm{w}_1}^{(\mathrm{L})}, \mathrm{h}_{\mathrm{w}_2}^{(\mathrm{L})}, \cdots, \mathrm{h}_{\mathrm{w}_\mathrm{m}}^{(\mathrm{L})}\right) \tag{10}$$

where $h_{w_j}^{(L)}$ denotes the refined node embedding generated by the L-th layer of the RGCN.

Attention Mechanism. Different pieces of evidence contribute differently to the claim. Therefore, we introduce an attention mechanism to weight various pieces of evidence. We calculate the attention weight α_i between evidence e_i and claimc. The formula is as follows:

$$\mathrm{\alpha}_\mathrm{i} = \frac{\mathrm{e}xp(Sim(\mathrm{h}_{\mathrm{e}_\mathrm{i}}, \mathrm{h}_\mathrm{c}))}{\sum_\mathrm{j} exp(Sim(\mathrm{h}_{\mathrm{e}_\mathrm{j}}, \mathrm{h}_\mathrm{c}))} \tag{11}$$

where Sim is a similarity function that captures the semantic relatedness between evidence and the claim. This allows the model to prioritize informative evidence while downplaying irrelevant or contradictory information. h_c represents the representation of the claim:

$$\mathrm{h}_\mathrm{c} = \mathrm{BERT}([\mathrm{CLS}]\ \mathrm{c}\ [\mathrm{SEP}]) \tag{21.1}$$

Veracity Prediction. The final veracity prediction is derived by integrating the attention-weighted evidence representations with the claim embedding. The prediction layer is formalized as:

$$\hat{\mathrm{y}} = \mathrm{softmax}(\mathrm{W}\ [\textstyle\sum_\mathrm{i} \mathrm{\alpha}_\mathrm{i}\, \mathrm{h}_{\mathrm{e}_\mathrm{i}}, \mathrm{h}_\mathrm{c}]) \tag{32}$$

where W denotes the classification weight matrix, and $\hat{y}$ presents the predicted probability distribution over the veracity labels.

Since the Claim Verification task involves three types of tags, the loss function is defined using categorical cross-entropy:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_i = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_i \left(-\sum_{k=0}^{2} y_{i,k}\, log(\hat{y}_{i,k})\right) \tag{43}$$

where N represents the number of samples. $\mathcal{L}_i$ is the loss for the i-th sample i. $y_{i,k}$ is the ground truth one-hot label vector for sample i, and $\hat{y}_{i,k}$ is the predicted probability for sample i belonging to class k.

3 Experiments

3.1 Dataset

Following Hu et al. [2], we evaluate our proposed model on CHEF dataset. The CHEF dataset is divided into 8,002 training samples, 999 development samples, and 999 test samples. A brief summary and corresponding detailed visualizations can be seen in Table 1.

Table 1. Dataset split sizes and statistics for CHEF.

Split	SUP	REF	NEI	Total
Train	22877	4399	776	8002
Dev	3333	333	333	999
Test	333	333	333	999
Avg #Words in the Claim				28
Avg #Words in the Google Snippets				68
Avg #Words in the Gold Evidences				126
Avg #Words in the Source Documents				3691

3.2 Setup

For the evidence selection component, we employ the F1 score to assess performance. In the claim verification phase, we follow the methodology proposed by Augenstein et al. [15] and use both Micro F1 and Macro F1 metrics hereafter referred to as Micro C-F1 and Macro C-F1, respectively to evaluate classification accuracy. These metrics are defined as follows:

$$Micro\ C-F1\ =\frac{2\sum_{i=1}^{C}P_iR_i}{\sum_{i=1}^{C}P_i+\sum_{i=1}^{C}R_i} \tag{54}$$

$$Macro\ C-F1\ =\frac{2}{C}\sum_{i=1}^{C}\frac{P_iR_i}{P_iR_i} \tag{65}$$

where C denotes the number of classification categories, and Pi, Ri represent the precision and recall for class i.

For the sentence encoder, we leverage the BERT-Base-Chinese pre-trained model [16] with a hidden dimension (d = 768). The GAT model is composed of 4 layers. The RGCN model is also configured with 4 layers. The AdamW optimizer [17] is employed with a learning rate of 4e-5, batch size of 32, and training for 40 epochs..

3.3 Baseline

Building on previous works [3, 18], we employ two types of baseline systems: Pipeline and Joint systems.

Pipeline Architectures sequentially perform evidence retrieval and claim verification:

Evidence Selection:

- **Google Snippets:** Leverage search engine previews from Google [5, 18].
- **Surface Ranker:** Employ TF-IDF to rank lexically similar sentences [5, 6, 19].
- **Semantic Ranker:** Utilize BERT pre-trained on Chinese corpora [16, 20] to compute cosine similarity between claim and document embeddings.
- **Hybrid Ranker:** Combine TF-IDF and BERT features via RankSVM for optimized ranking [21].

Table 2. Results of FGGNN and baseline models on CHEF.

System/Evidence			Test Set		Dev Set	
			Micro C-F1	Macro C-F1	Micro C-F1	Macro C-F1
Pipeline		No Evidence	54.46	52.49	54.76	52.97
		Google Snippets	62.07	60.61	62.31	60.87
		Surface Ranker	63.17	61.47	63.53	61.78
		Semantic Ranker	63.47	61.94	63.73	62.42
		Hybrid Ranker	63.29	61.80	63.12	61.53
Joint	Reinforce	Google Snippets	63.76	61.74	63.54	61.48
		Source Documents	64.37	62.46	64.68	62.63
	Multi-task	Google Snippets	62.78	61.98	62.49	62.37
		Source Documents	65.02	63.12	65.41	63.38
	Latent	Google Snippets	64.45	62.52	64.74	62.80
		Source Documents	66.77	64.65	66.96	64.92
Pipeline	Reread	Source Documents	70.87	68.78	71.31	69.25
	FGGNN	Source Documents	**72.74**	**70.57**	**73.15**	**71.11**
Pipeline		Gold evidence	78.99	77.62	79.84	78.47

- **ReRead:** Apply plausibility, completeness, and sufficiency criteria to select evidence from real-world documents [22].

Claim Verification:

- **BERT-Based Model:** Use BERT embeddings fed into a multi-layer perceptron [19, 23].

Joint systems consider evidence extraction as a latent factor and optimize the evidence extraction procedure in conjunction with the loss of claim verification.

- **Reinforce System:** It uses REINFORCE [24] to optimize the evidence retriever.
- **Multi-task System:** Simultaneously predict evidence subsets and claim veracity, minimizing the combined loss [6].
- **Latent System:** Employ the Hard Kumaraswamy distribution to sample evidence selectors [3, 25].

Baseline Controls:

- **No Evidence:** Represents performance without external evidence.
- **Gold Evidence:** Establishes an upper bound using oracle-selected evidence.

We compare our proposed FGGNN with various baselines. Table 2 reports the average performance across 5 training/testing runs on the CHEF dataset's development and test splits [3]. Table 3 evaluates the quality of retrieved evidence. Key findings include:

(1) Extracting supporting sentences from full-text documents yields higher C-F1 scores compared to using Google snippets alone, as source documents contain richer contextual information.

(2) Joint optimization frameworks generally outperform pipeline architectures as it concurrently optimizes evidence selection and claim verification in fact verification. However, the FGGNN model surpasses even these advanced Joint models. On the Dev set, FGGNN reaches an E-F1 of 96.6% and a Micro C-F1 of 73.15%. Compared to the Latent Joint model, FGGNN has 6.1% and 6.19% higher scores in these two metrics. This shows FGGNN is better at evidence selection and can improve claim verification accuracy via more effective evidence integration.

(3) FGGNN achieves a Micro C-F1 score of 73.15% and a Macro C-F1 score of 71.11% on Dev set outperforming all other models. Compared with ReRead [22], FGGNN demonstrates stronger generalization ability in handling complex evidence and exhibits a more balanced performance across various label classification tasks.

Table 3. Quality of Retrieved Evidence Analysis.

Methods	Test E-F1	Dev E-F1
Surface	85.3	84.6
Semantic	88.1	88.4
Hybrid	87.7	87.5
Reinforce	89.6	89.3
Multi-task	90.4	90.3
Latent	90.8	90.5
ReRead	95.3	95.1
FGGNN	**96.5**	**96.6**

3.4 Ablation Analysis

To validate the contributions of individual components, we perform the following experiments: 1) w/o Gold: During evidence selection, replace gold-standard evidence with document-level summaries to assess the impact of precise evidence. 2) w/o wg: Remove the word-level evidence graph, using only sentence-level representations to evaluate structural granularity. 3) w/o am: Replace the attention mechanism with convolution during claim verification to isolate the effect of adaptive weighting.

The ablation study findings are presented in Table 4. By examining these results, we can notice that the model without the supervision of gold evidence experiences a 1.84% reduction in Micro C-F1. This drop indicates that gold evidence plays a vital role in the evidence selection process. After removing the word-level evidence graph, the performance of the model also declined accordingly. This result suggests that fine-grained modeling of evidence is beneficial. It can better capture semantic relationships across sentences. This in turn enhances the overall performance of the model. Instead of using the attention mechanism, we implementing convolution also results in a performance drop of 0.89% and 0.68%. The attention mechanism can prioritize relevant information. This helps with a better understanding of claims and associated evidence.

Table 4. The analysis of ablation on FGGNN. The arrow ↓ represents the decrease

Methods	Micro C-F1	Macro C-F1
FGGNN	73.15	71.11
w/o Gold	↓1.84	↓1.71
w/o wg	↓1.63	↓1.92
w/o am	↓0.89	↓0.68

4. Conclusion

In this study, we proposed the Fine-Grained Graph Neural Network (FGGNN) for fact verification. The evidence selection module, based on a sentence-level graph, uses GNNs and attention mechanism to precisely locate key evidence. The claim verification module captures fine-grained relationships and weights crucial evidence through word-level evidence graphs and RGCNs. Empirical evaluations on standard benchmark datasets demonstrate that FGGNN significantly outperforms all baseline models, confirming its effectiveness and superiority in CDFV.

Acknowledgements. The authors would like to thank the two anonymous reviewers for their comments on this paper. This research was supported by the National Natural Science Foundation of China (Nos. 62376181 and 62276177) and the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (No. 24KJB520036).

References

1. Schlichtkrull, M., Kipf, T. N., Bloem, P., van den Berg, R., Titov, I., & Welling, M. (2018). Modeling relational data with graph convolutional networks. ESWC 2018, Heraklion, Crete, Greece, pages 593-607.
2. Andreas Vlachos and Sebastian Riedel. Fact checking: Task definition and dataset construction. In Workshop on Language Technologies and Computational Social Science, Baltimore, MD, USA, pages 18–22, 2014.
3. Xuming Hu, Zhijiang Guo, GuanYu Wu, Aiwei Liu, Lijie Wen, and Philip S Yu. Chef: A pilot chinese dataset for evidence-based fact-checking. In Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Seattle, WA, United States, pages 3362–3376, 2022.
4. Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading Wikipedia to answer open-domain questions. In Annual Meeting of the Association for Computational Linguistics, Vancouver, Canada, pages 1870–1879, 2017.
5. Rami Aly, Zhijiang Guo, Michael Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. Feverous: Fact extraction and verification over unstructured and structured information. In Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, virtual, 2021.

6. James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. Fever: a large-scale dataset for fact extraction and verification. In Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, Louisiana, USA, pages 809–819, 2018.
7. Nan Hu, Zirui Wu, Yuxuan Lai, Xiao Liu, and Yansong Feng. Dual-channel evidence fusion for fact verification over texts and tables. In Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Seattle, WA, United States, pages 5232–5242, 2022.
8. Mostafa Bouziane, Hugo Perrin, Amine Sadeq, Thanh Nguyen, Aurélien Cluzeau, and Julien Mardas. Fabulous: fact-checking based on understanding of language over unstructured and structured information. In Workshop on Fact Extraction and VERification, Dominican Republic, pages 31–39, 2021.
9. Chen Zhao, Chenyan Xiong, Corby Rosset, Xia Song, Paul Bennett, and Saurabh Tiwary. Transformer-xh: Multi-evidence reasoning with extra hop attention. In International Conference on Learning Representations, Addis Ababa, Ethiopia, 2020.
10. Junfei Wu, Weizhi Xu, Qiang Liu, Shu Wu, and Liang Wang. Adversarial contrastive learning for evidence-aware fake news detection with graph neural networks. IEEE Transactions on Knowledge and Data Engineering, 36(11):5591–5604,2023.
11. Han Cao, Lingwei Wei, Mengyang Chen, Wei Zhou, and Songlin Hu. Are large language models good fact checkers: A preliminary study. CoRR, abs/2311.17355, 2023.Author, F.: Article title. Journal **2**(5), 99–110 (2016) Author, F., Author, S.: Title of a proceedings paper. In: Editor, F., Editor, S. (eds.) CONFERENCE 2016, LNCS, vol. 9999, pp. 1–13. Springer, Heidelberg (2016)
12. Xuan Zhang and Wei Gao. Towards llm-based fact verification on news claims with a hierarchical step-by-step prompting method. In International Joint Conference on Natural Language Processing and Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, Nusa Dua, Bali, pages 996–1011, 2023.
13. Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. In-context retrieval-augmented language models. Transactions of the Association for Computational Linguistics, 11:1316–1331, 2023.
14. Jie Zhou, Xu Han, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Gear: Graph-based evidence aggregating and reasoning for fact verification. In Proceedings of the Conference of the Association for Computational Linguistics, Florence, Italy, 2019.
15. Isabelle Augenstein, Christina Lioma, Dongsheng Wang, Lucas Chaves Lima, Casper Hansen, Christian Hansen, and Jakob Grue Simonsen. Multifc: A real-world multi-domain dataset for evidence-based fact checking of claims. In EMNLP and IJCNLP, Hong Kong, China, pages 4684–4696, 2019.
16. J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, pages 4171–4186, 2019.
17. Nicolas Dintzner, Arie van Deursen, and Martin Pinzger. Fever: An approach to analyze feature-oriented changes and artefact co-evolution in highly configurable systems. Empirical Software Engineering, 23:905–952, 2018.
18. Ashim Gupta and Vivek Srikumar. X-fact: A new benchmark dataset for multilingual fact checking. In Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing, Virtual Event, pages 675–682, 2021.

19. Yichen Jiang, Shikha Bordia, Zheng Zhong, Charles Dognin, Maneesh Singh, and Mohit Bansal. Hover: A dataset for many-hop fact extraction and claim verification. In Findings of the ACL, Online Event, pages 3441–3460, 2020.
20. Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In EMNLP, Online, pages 38–45, 2020.
21. Shaden Shaar, Giovanni Da San Martino, Nikolay Babulkov, and Preslav Nakov. That is a known lie: Detecting previously fact-checked claims. In Annual Meeting of the Association for Computational Linguistics,Online, pages 3607–3618, 2020.
22. Xuming Hu, Zhaochen Hong, Zhijiang Guo, Lijie Wen, and Philip Yu. Read it twice: Towards faithfully interpretable fact verification by revisiting evidence. In International ACM SIGIR Conference on Research and Development in Information Retrieval, Taipei, Taiwan, pages 2319–2323, 2023.
23. Tal Schuster, Adam Fisch, and Regina Barzilay. Get your vitamin c! robust fact verification with contrastive evidence. In Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online, 2021.
24. Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8:229–256, 1992.
25. Jasmijn Bastings, Wilker Aziz, and Ivan Titov. Interpretable neural predictions with differentiable binary variables. In Proceedings of the 57th Conference of the Association for Computational Linguistics, Florence, Italy, pages 2963–2977, 2019.