Improving Event Detection via Trigger Word Expansion

1st Qing Zhou
School of Computer and Information
Anhui Polytechnic University
Wuhu, China
qingzhou1102@gmail.com

2nd Subin Huang*
School of Computer and Information
Anhui Polytechnic University
Wuhu, China
subinhuang@ahpu.edu.cn

3th Chengzhen Yu
School of Computer and Information
Anhui Polytechnic University
Wuhu, China
zasetyubokerse@gmail.com

4rd Daoyu Li
School of Computer and Information
Anhui Polytechnic University
Wuhu, China
lidaoyu0801@gmail.com

5th Junjie Chen
School of Computer and Information
Anhui Polytechnic University
Wuhu, China
jorji.chen@gmail.com

6th Sanmin Liu
School of Computer and Information
Anhui Polytechnic University
Wuhu, China
sanmin.liu@ahpu.edu.cn

***Abstract*—Event detection is a fundamental task in information extraction, aiming to identify trigger words from given texts and categorize them into distinct event types. Existing approaches for event detection predominantly rely on annotating trigger words to classify events, which can lead to semantic recognition errors due to the oversimplified semantics of these trigger words. To tackle this challenge, we propose a trigger word expansion-based approach for robust contextual event detection. Specifically, we propose a framework comprising a trigger word extractor within a model for trigger word expansion classifier. This enables event detection to be conducted through trigger word expansion, enriching the semantics of trigger words. Additionally, we leverage the GPT model to generate contexts for the expanded trigger words, thereby enhancing the contextual understanding of trigger words. Extensive experiments conducted on the standard MAVEN benchmark dataset showcase the superior performance of our approach compared to state-of-the-art methods. This confirms the effectiveness of our proposed approach over existing approaches that rely on trigger word annotation.**

***Index Terms*—Event detection, trigger word expansion, GPT**

I. Introduction

Event detection is a key task in the fields of information extraction [1]–[3] and natural language processing [4], focused on identifying trigger words within provided texts and categorizing them into specific event types. As illustrated by sentences S1 and S2 in Figure 1, this scenario typifies normal cases of event detection task. Such tasks are extensively applied across a multitude of domains, including but not limited to news aggregation and summarization, social media monitoring, and financial market analysis [5], [6].

Traditionally, event detection comprises two main steps: identifying trigger words, referred to as Trigger Identification (TI), and classifying them into specific event types, known as Trigger Classification (TC). As shown in the badcase in Figure 1, the two steps of event detection still reveal two main issues, despite significant progress achieved by deep learning in event detection tasks.

- Simplistic semantics of trigger words. In the TI stage, how to address the problem of overly simplistic semantics for the same type of trigger words within a sentence.
- Context of trigger words. In the TC stage, how can we augment the context of trigger words to ensure their precise classification into specific event types.

The above issues indicate that a trigger word has multiple meanings and the same trigger word in different contexts may represent inconsistent event types.

To tackle the first challenge, we propose a framework comprising a trigger word extractor within a model for trigger word expansion classifier. This enables event detection to be conducted through trigger word expansion, enriching the semantics of trigger words. To address the second issue, we

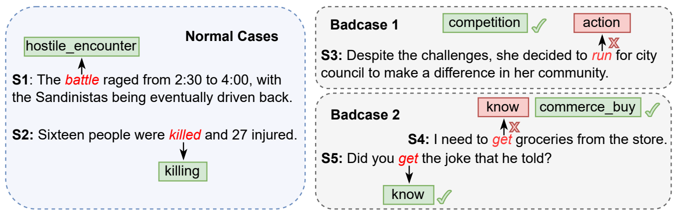

Normal Cases
hostile_encounter
S1: The battle raged from 2:30 to 4:00, with the Sandinistas being eventually driven back.
S2: Sixteen people were killed and 27 injured.
killing
Badcase 1
competition
action
S3: Despite the challenges, she decided to run for city council to make a difference in her community.
Badcase 2
know
commerce_buy
S4: I need to get groceries from the store.
S5: Did you get the joke that he told?
know

Fig. 1. Task illustration. In sentence $S1$, the annotated trigger word ‘battle’ is used for trigger identification. The trigger word ‘battle’ is classified into the ‘hostile_encounter’ event type. Similarly, in sentence $S2$, the trigger word is ‘killed’, which is classified into the ‘killing’ event type. However, a simple trigger word can indeed lead to semantic drift issues in event detection. For example, in sentence $S3$, the trigger word ‘run’ is incorrectly classified into the ‘action’ event type, whereas its correct meaning is ‘campaign’, corresponding to the ‘competition’ event type. In sentences $S4$ and $S5$, the annotated trigger word is ‘get’ for both, corresponding to the ‘know’ event type. However, the correct event type for S4 is ‘commerceb_buy’.

This work is funded by the Anhui Provincial Natural Science Foundation with Grant No. 2308085MF220 and co-supported by the Anhui University Natural Science Foundation with Grant Nos. 2022AH050972 and KJ2021A0516.

* Subin Huang is the corresponding author.

DOI reference number: 10.18293/SEKE2024-030

utilize the GPT [7] model to generate contexts for each word in the expanded set of synonymous trigger words, thereby enriching the semantics of the annotated trigger words and enhancing their context. This approach enriches the original annotated trigger words with different contexts under the similar semantics, allowing the trigger words to be accurately classified into specific event types.

To confirm the effectiveness of our proposed approach, we evaluated our model on the MAVEN [8] benchmark dataset, confirming that our approach outperforms existing methods that are solely based on trigger word annotations.

In summary, we present three principal contributions:

- We propose a framework for a trigger word extractor and develop a model within it, called a trigger word expansion classifier, enabling the effective expansion of trigger words into a set of words with similar semantics.
- Using the GPT model, we generate contexts for each word in the expanded set of trigger words, enhancing the contextual understanding of trigger words and thereby enabling accurate classification of trigger words into specific event types.
- We conduct comprehensive experiments using the MAVEN benchmark dataset to show that our approach surpasses existing state-of-the-art methods and to highlight the efficacy of trigger word expansion for robust contextual event detection.

II. Related Works

Event Detection. Event detection is a critical task within the field of natural language processing. In traditional event detection, early strategies mainly revolved around identifying and tagging event expressions by integrating linguistic and statistical data through relatively straightforward methods. In the initial phases of event detection tasks, feature engineering, supervised learning, and traditional information retrieval techniques [9], [10] were commonly employed. However, with the advent of deep learning, the field of event detection has increasingly favored methods that rely primarily on deep learning techniques. Deep learning has enabled end-to-end learning approaches for event detection tasks, which simplify the task process by directly learning event features and relationships from raw text data [11], [12].

Trigger Word Application. Trigger words play an essential role in event detection tasks. In the task of event detection, accurately annotating and categorizing trigger words into specific event types is essential for grasping the dynamics of events. Initially, trigger annotation mainly employed the clustering method, utilizing self-similarity to converge the K-value in the K-means algorithm guided by event triggers, and optimizing clustering algorithms [13]. With the development of deep learning, researchers are focused on utilizing the capabilities of Convolutional Neural Networks to automatically extract higher-level features, facilitating the exploration of correlations between triggers and event types. These approaches assist in accurately locating trigger words and mitigating their ambiguity issues. While progress has been made in trigger word annotation, the semantics of annotated trigger words in texts can often be overly specific, affecting event detection accuracy. To address this challenge, we propose a trigger word expansion-based approach for robust contextual event detection.

III. Method

Figure 2 shows the three key steps in our proposed approach.

- Trigger word expansion. A trigger word expansion extractor framework is proposed for trigger word expansion. The extractor transforms a single trigger word into a collection of words with similar meanings, enriching the semantics of the original trigger words.
- GPT-generated context. Using the GPT model, we generate contexts for each word in the expanded set of trigger words, enhancing the contextual understanding of trigger words, accurately classifying trigger words into specific event types.
- Event type prediction. We utilize the OmniEvent [14] framework for predicting event types in sentences. Specifically, we train a model for event detection within the OmniEvent framework using the MAVEN dataset. Subsequently, we input the original sentence along with the generated contexts into the model as a collection of sentences to predict the event type.

A. Trigger Word Expansion

1) Data preparation: Before training the trigger word expansion classifier model, preliminary data preparation is conducted. Initially, a series of data processing steps are undertaken to extract relevant word corpora S from the MAVEN dataset.

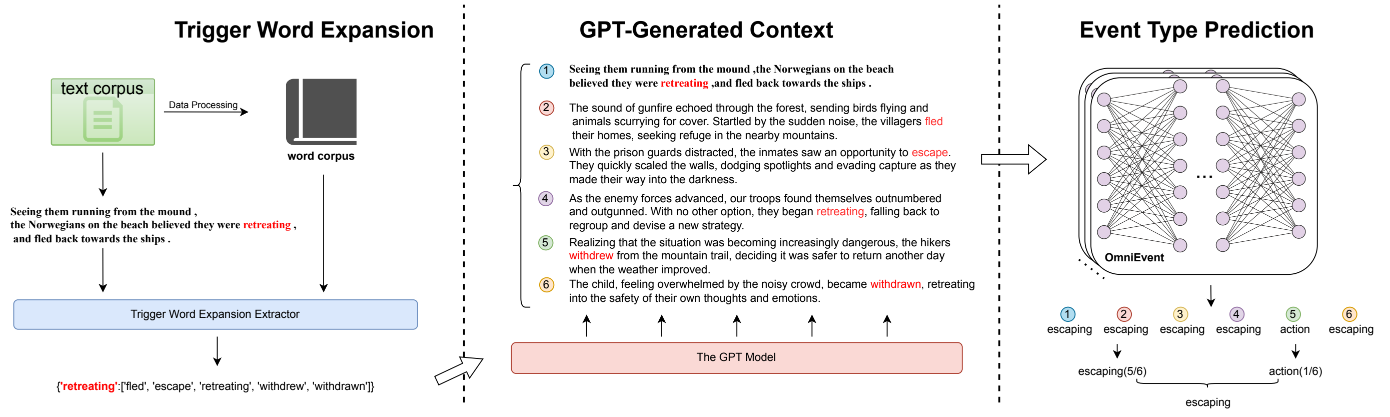
$$\begin{aligned} S &= \{S_1, S_2, S_3, S_4, \dots\}, \\ S_1 &= \{event\_type : triggers\}, \\ triggers &= \{trigger_1, trigger_2, trigger_3, \dots\}. \end{aligned} \tag{1}$$

For example, S_1 is a subset of S:

$$\begin{aligned} S_1 = \{&`know' : \{`observed', \\ &`perceived', `spotted', `realize', \dots\}\}, \end{aligned} \tag{2}$$

where ‘know’ represents the event type, and ‘observed’, ‘perceived’, etc., are the associated trigger words for the ‘know’ event type. Subsequently, the construction of the training and testing datasets is as follows.

- **Construction of positive samples**. For all subsets in word corpora S, positive samples are formulated in the form of triple $(event\ type, related\ trigger\ word, label)$. Taking Equation (2) as an example, positive samples appear as (‘know’, ‘observed’, 1), (‘know’, ‘perceived’, 1), etc., where ‘know’ is the event type in S_1, and ‘observed’, ‘perceived’, etc., are the triggers listed in the S_1. Similarly, the construction of positive samples for S_2, S_3, etc., follows the same process.



**Seeing them running from the mound ,
the Norwegians on the beach believed they were retreating ,
and fled back towards the ships .**

{'**retreating**':['fled', 'escape', 'retreating', 'withdrew', 'withdrawn']}

1. **Seeing them running from the mound ,the Norwegians on the beach believed they were retreating ,and fled back towards the ships .**
2. The sound of gunfire echoed through the forest, sending birds flying and animals scurrying for cover. Startled by the sudden noise, the villagers fled their homes, seeking refuge in the nearby mountains.
3. With the prison guards distracted, the inmates saw an opportunity to escape. They quickly scaled the walls, dodging spotlights and evading capture as they made their way into the darkness.
4. As the enemy forces advanced, our troops found themselves outnumbered and outgunned. With no other option, they began retreating, falling back to regroup and devise a new strategy.
5. Realizing that the situation was becoming increasingly dangerous, the hikers withdrew from the mountain trail, deciding it was safer to return another day when the weather improved.
6. The child, feeling overwhelmed by the noisy crowd, became withdrawn, retreating into the safety of their own thoughts and emotions.

Fig. 2. Overview framework

- **Construction of negative samples**. For all subsets in word corpora S, negative samples are initially generated from a negative sample pool. For example, when creating negative samples for S_1, the negative sample pool is generated as $NSP = \{S(triggers) - S_1(triggers)\}$, where $S(triggers)$ refers to all triggers data within S, and $S_1(triggers)$ refers to all triggers data within S_1. This implies that the NSP comprises all trigger word corpora excluding those found within S_1. The construction of negative samples for S_1 proceeds as ('know', 'tsunami', 0), where 'know' is the event type in S_1, and 'tsunami' is a randomly selected trigger from NSP. Likewise, the construction of negative samples for S_2, S_3, etc., adopts the same process.

2) Training Trigger Word Expansion Classifier: As illustrated in Figure 3, inspired by SynSetMine [15], this paper proposes a trigger word expansion classifier to mine additional trigger words [16], where the classifier receives an event type and a trigger word as input and outputs a score to determine whether these two words belong to the same category.

In the architecture of the trigger word expansion classifier, we adopt a model based on BERT [17] and append a classifier to interpret the output. The classifier can obtain a score using a sigmoid function $sigmoid(x) = (1 + e^{-x})^{-1}$ to determine whether two inputs belong to the same category. We utilize a cross-entropy loss function to train the trigger word expansion classifier as shown in Equation (3).

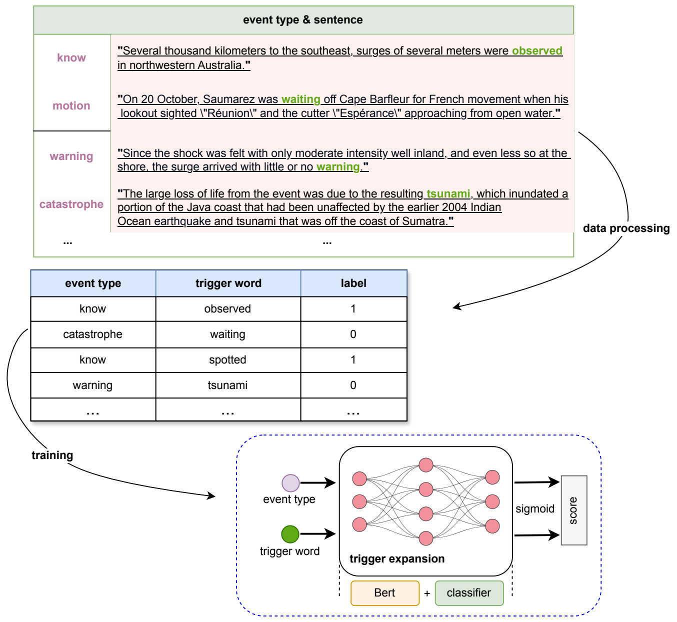
$$L = -\sum y_i \log(p_i) + (1 - y_i) \log(1 - p_i), \qquad (3)$$

where y_i is a binary indicator (0 or 1). p_i is the output of the trigger word expansion classifier.

3) Trigger Word Set Expansion: In Algorithm 1, we delineate the detailed process of generating trigger word sets. We input a sentence with marked trigger words (W_1) and a processed word corpus into the trigger word expansion extractor to accomplish the set generation. Here, the word corpus constitutes a dictionary composed of all event types from the word corpora S. For the trigger word expansion extractor, we extract the marked trigger word W_1 and iteratively feed it, along with words from the word corpus, into our trained trigger word expansion classifier. Specifically, suppose the word corpus(WC) contains et_1, et_2, et_3, ... , et_n; we input W_1 and each event type from the word corpus into the trigger expansion classifier in turn, and record the maximum output score ($ScoreMax$) and the corresponding event type (et_i). Subsequently, we randomly select five trigger words ($trigger^1, trigger^2, ..., trigger^5$) from the mapping list ($mapped\ triggers$) of event types and trigger words as the expanded trigger words for W_1.

$$mapped\ triggers = S_i[et_i] \qquad (4)$$

where S_i is a subset of S.



event type & sentence	
know	"Several thousand kilometers to the southeast, surges of several meters were **observed** in northwestern Australia."
motion	"On 20 October, Saumarez was **waiting** off Cape Barfleur for French movement when his lookout sighted \"Réunion\" and the cutter \"Espérance\" approaching from open water."
warning	"Since the shock was felt with only moderate intensity well inland, and even less so at the shore, the surge arrived with little or no **warning**."
catastrophe	"The large loss of life from the event was due to the resulting **tsunami**, which inundated a portion of the Java coast that had been unaffected by the earlier 2004 Indian Ocean earthquake and tsunami that was off the coast of Sumatra."
...	...

event type	trigger word	label
know	observed	1
catastrophe	waiting	0
know	spotted	1
warning	tsunami	0
...	...	...

Fig. 3. Training trigger word expansion classifier

Algorithm 1: Set Generation Algorithm

```
Input: A trigger word expansion classifier $f$; An input vocabulary word corpus including all event types: $WC = \{et_1, et_2, et_3, \ldots\}$; A marked trigger words $W_1$; The mapping list: $mapped\ triggers$; An empty list: $el$
Output: The trigger expansion sets: $TES = \{W_1 : [trigger^1, trigger^2, trigger^3, trigger^4, trigger^5]\}$
1  $TES \leftarrow \{W_1 : []\}$;
2  for $i$ in $WC$ do
3      $ScoreMax \leftarrow 0$;
4      $Score \leftarrow f(W_1, i)$;
5      if $Score > Score\ max$ then
6          $ScoreMax \leftarrow Score$;
7          $record\ the\ corresponding\ event\ type\ et_i$
8      end
9  end
10 $el \leftarrow$ random.sample($mapped\ triggers, 5$);
11 $TES[W_1]$.append($el$);
12 return $TES$;
```

B. GPT-Generated Context

In this paper, we utilize the GPT model, denoted as GPT, to generate contexts for trigger words, with each trigger word generating ten contexts. The following details the process of context generation utilizing GPT:

- **Generating context sentences.** The trigger words in trigger expansion set TES are sequentially fed into GPT, resulting in the generation of context sentences $SC = \{SC_{trigger^1}, SC_{trigger^2}, \ldots, SC_{trigger^5}\}$, $SC_{trigger^i} = \{s_1, s_2, \ldots, s_{10}\}$.

$$TES = \{W_1 : [trigger^1, trigger^2, trigger^3, trigger^4, trigger^5]\} \tag{5}$$

$$SC = GPT(TES). \tag{6}$$

- **Evaluating context sentences.** We evaluate whether the selected SC meets the standards by comparing the cosine similarity between the original sentence sW_1 and SC.

$$sim(sW_1, s_i) = \frac{sW_1 \cdot s_i}{\|sW_1\|\|s_i\|}, \tag{7}$$

where $s_i \in SC$. We select the top generated contexts $\hat{SC} = \{\hat{s_1}, \hat{s_2}, \ldots, \hat{s_5}\}$ from $SC_{trigger^i}$. Ultimately, the contexts ctx for trigger words are generated and denoted as follows.

$$ctx = \{sW_1, \hat{SC}\}. \tag{8}$$

C. Event Type Prediction

In this paper, we employ the OmniEvent (OE) framework to predict event types for the connected sentences in ctx. Initially, the sentences in ctx are sequentially input into OE framework, and the event type E_i for each sentence is recorded.

$$E_i = OE(ctx_i). \tag{9}$$

Subsequently, we utilize the distribution proportions of the expected types for each context to predict the event type, denoted as:

$$P(E_i) = \frac{Num(E_i)}{Num}, \tag{10}$$

$$Event\ Type = \arg\max_{E_i} P(E_i), \tag{11}$$

where E_i represents the event type predicted by OE framework. $Num(E_i)$ represents the number of sentences of event type E_i. Num represents the total number of sentences.

IV. Experiments

A. Dataset

Our approach is assessed on the MAVEN dataset, derived from English Wikipedia and FrameNet [18]. Table I presents the specific data statistics for MAVEN. To fulfill our requirements, we have constructed experimental samples that conform to these specifications. Table II provides the detailed statistics of our experimental sample data.

TABLE I
MAVEN DATASET STATISTICS

Docs	Sents	Event type	Instances
4480	49873	168	118732

TABLE II
EXPERIMENT SAMPLE STATISTICS

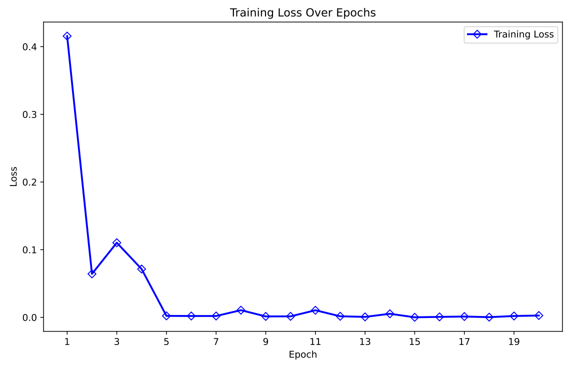
	train samples	test samples	all samples
Trigger Word Expansion	10,962	2,741	13,703
Event Type Prediction	2,913	857	3,770

B. Experimental Setup

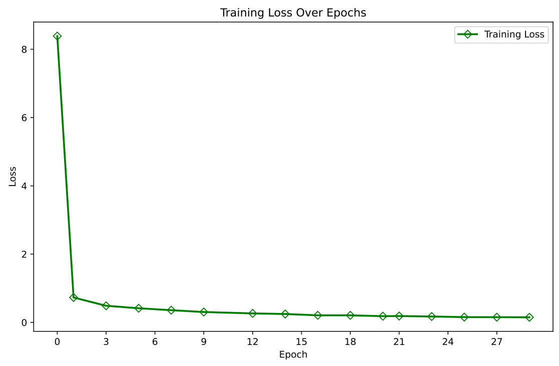
In this paper, we organize our approach into three distinct modules: Trigger Word Expansion (TWE), GPT-Generated Context (GPTGC), and Event Type Prediction (ETP). The specific model parameters relevant to our experiments are detailed in Table III.

TABLE III
EXPERIMENT PARAMETERS STATISTICS

	TWE	GPTGC	ETP
backbone	BERT	GPT2	T5
hidden size	768	768	768
learning rate	1.0e-4	2.0e-5	5.0e-5
optimizer	adamw	gelu	relu



(a) Training loss per epoch in trigger word expansion classifier.



(b) Training loss per epoch in event type prediction.

Fig. 4. Model performance.

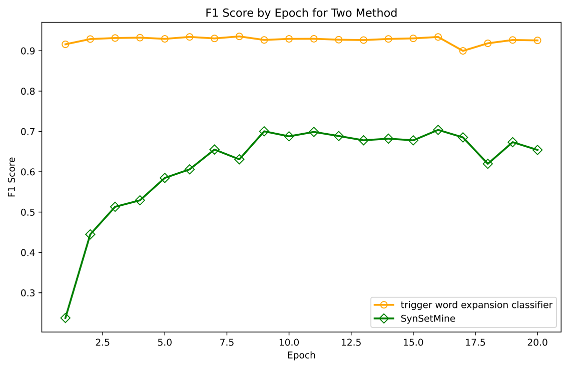


Fig. 5. F1 score of trigger word expansion classifier vs SynSetMine.

C. Baselines and Metrics

To validate our proposed approach, we choose the following models as the baselines.

- **DMCNN [19]:** A two-phase approach that first utilizes CNN as the foundation, deriving features through a dynamic multi-pool convolutional neural network focusing on local features, and then assesses the relationship between the text's overall structure and its components.
- **HBTNGMA [20]:** A dual-stage approach is employed, initially incorporating a multi-layer attention mechanism utilizing sticky notes and gated units to fuse dynamic information within sentences, followed by the application of this framework for event detection in sentences.
- **MLBiNet [21]:** A method is implemented where cross-sentence semantic information and event details were extracted using a multi-layer bidirectional network.
- **MOGANED [22]:** A syntactic dependency tree enhanced with GCN was employed, consolidating multi-layer syntactic information within sentences via an attention mechanism.

To comprehensively assess the model's performance in the TWE and ETP modules, Precision (P), Recall (R), and the F1 Score (F1) are employed as the primary metrics for performance evaluation. Equations (12) to (14) are as follows.

$$P = \frac{TP}{TP + FP} \tag{12}$$

$$R = \frac{TP}{TP + FN} \tag{13}$$

$$F1 = 2 \times \frac{P \times R}{P + R} \tag{14}$$

Where TP represents the number of instances correctly predicted as positive by the model; FP represents the number of negative instances incorrectly predicted as positive by the model; FN represents the number of positive instances incorrectly predicted as negative by the model.

D. Results and Analysis

Performance of the trigger word expansion classifier. Figure 4a and Figure 5 are the experimental results for the trigger word expansion classifier. As shown in Figure 4a, we can observe that, with an increase in epochs, there is a reduction in loss. Figure 5 presents a comparison of F1 scores across epochs for two different methodologies: a trigger word expansion classifier and SynSetMine. The trigger word expansion classifier consistently maintains an F1 score above 0.9 throughout all epochs. In contrast, SynSetMine exhibits a gradual improvement in F1 score and peaks just below 0.7 around epoch 10. Beyond this peak, the F1 score levels off with minor fluctuations, finishing slightly below its peak performance by epoch 20. The graph distinctly highlights the superior and consistent performance of the trigger word expansion classifier in terms of F1 scores compared to SynSetMine.

Performance of the event type prediction. Figure 4b and Table IV are the experimental results for the event type prediction. As shown in Figure 4b, we can observe a rapid decline in loss during the initial epochs, notably experiencing a steep drop within the first two epochs. After this initial drop, the rate of loss reduction slows down and gradually stabilizes.

TABLE IV
OVERALL PERFORMANCE

Methods	**P**	**R**	**F1**
DMCNN	**66.3**	55.9	60.6
HBTNGMA	63.3	62.6	62.7
MLBiNet	63.5	63.8	63.6
MOGANED	63.4	64.1	63.8
Ours	64.9	**66.8**	**65.8**

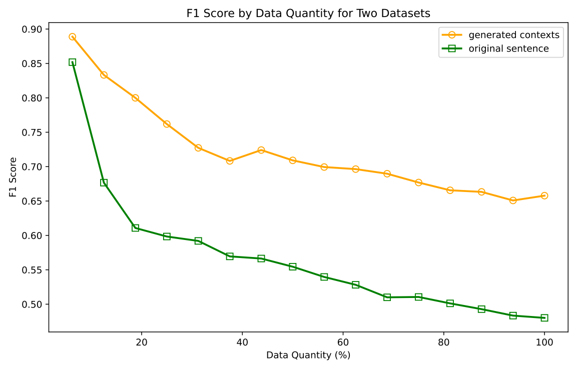


Fig. 6. F1 score of generated contexts vs original sentence.

As shown in Table IV, we compare the overall performance of various methods across three metrics. Our method outperforms other methods in terms of recall and F1 score. Moreover, Figure 6 presents a comparison of the F1 scores for two datasets in the event type evaluation using the OmniEvent model. The results indicate that the F1 scores for event type prediction with generated contexts consistently outperform those with original sentence. With the increase in data quantity, the F1 scores for generated contexts show a gradual decline before stabilizing, whereas the scores for original sentence decrease initially, reach a plateau, and then exhibit a slight continuing decline. Overall, the model demonstrates superior performance in event type evaluation when utilizing generated contexts compared to original sentences.

V. Conclusion

In this paper, we delve into leveraging annotated trigger words within sentences to improve the accuracy of detecting corresponding event types. We introduce a trigger word expansion-based approach for robust contextual event detection. Within this framework, we effectively expand trigger words and utilize the GPT model to generate contexts for these expanded triggers. Subsequently, we predict event types of sentence collections using a model trained via the OmniEvent framework. The effectiveness of our proposed approach is validated through experiments across multiple models on the MAVEN dataset.

References

[1] D. Ahn, “The stages of event extraction,” in *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, 2006, pp. 1–8.

[2] W. Liu, Z. Yang, Y. Cao, and J. Huo, “Discovering the influences of the patent innovations on the stock market,” *Information Processing & Management*, vol. 59, no. 3, p. 102908, 2022.

[3] B. Li, G. Fang, Y. Yang, Q. Wang, W. Ye, W. Zhao, and S. Zhang, “Evaluating chatgpt’s information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness,” *arXiv preprint arXiv:2304.11633*, 2023.

[4] E. Alomari, I. Katib, and R. Mehmood, “Iktishaf: A big data road-traffic event detection tool using twitter and spark machine learning,” *Mobile Networks and Applications*, vol. 28, no. 2, pp. 603–618, 2023.

[5] S. Zhao, Y. Gao, G. Ding, and T.-S. Chua, “Real-time multimedia social event detection in microblog,” *IEEE transactions on cybernetics*, vol. 48, no. 11, pp. 3218–3231, 2017.

[6] Q. Li and Q. Zhang, “A unified model for financial event classification, detection and summarization,” in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 2021, pp. 4668–4674.

[7] P. Budzianowski and I. Vulić, “Hello, it’s gpt-2–how can i help you? towards the use of pretrained language models for task-oriented dialogue systems,” *arXiv preprint arXiv:1907.05774*, 2019.

[8] X. Wang, Z. Wang, X. Han, W. Jiang, R. Han, Z. Liu, J. Li, P. Li, Y. Lin, and J. Zhou, “Maven: A massive general domain event detection dataset,” *arXiv preprint arXiv:2004.13590*, 2020.

[9] K. Cao, X. Li, and R. Grishman, “Improving event detection with dependency regularization,” in *Proceedings of the International Conference Recent Advances in Natural Language Processing*, 2015, pp. 78–83.

[10] Y. Qin, Y. Zhang, M. Zhang, and D. Zheng, “Feature-rich segment-based news event detection on twitter,” in *Proceedings of the sixth international joint conference on natural language processing*, 2013, pp. 302–310.

[11] J. W. Orr, P. Tadepalli, and X. Fern, “Event detection with neural networks: A rigorous empirical evaluation,” *arXiv preprint arXiv:1808.08504*, 2018.

[12] W. Liu, S. Li, Y. Cao, and Y. Wang, “Multi-task learning based high-value patent and standard-essential patent identification model,” *Information Processing & Management*, vol. 60, no. 3, p. 103327, 2023.

[13] X. Zhang, B. Li, and Y. Tian, “Self-similarity clustering event detection based on triggers guidance,” in *Web Information Systems and Mining: International Conference, WISM 2009, Shanghai, China, November 7-8, 2009. Proceedings*. Springer, 2009, pp. 63–70.

[14] P. Hao, W. Xiaozhi, Y. Feng, Z. Kaisheng, H. Lei, L. Juanzi, L. Zhiyuan, and S. Weixing, “The devil is in the details: On the pitfalls of event extraction evaluation,” *arXiv preprint arXiv:2306.06918*, 2023.

[15] J. Shen, R. Lyu, X. Ren, M. Vanni, B. M. Sadler, and J. Han, “Mining entity synonyms with efficient neural set generation,” *CoRR*, vol. abs/1811.07032, 2018. [Online]. Available: http://arxiv.org/abs/1811.07032

[16] S. Huang, Y. Xiu, J. Li, S. Liu, and C. Kong, “A bilateral context and filtering strategy-based approach to chinese entity synonym set expansion,” *Complex & Intelligent Systems*, vol. 9, no. 5, pp. 6065–6085, 2023.

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[18] C. F. Baker, C. J. Fillmore, and J. B. Lowe, “The berkeley framenet project,” in *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*, 1998.

[19] Y. Chen, L. Xu, K. Liu, D. Zeng, and J. Zhao, “Event extraction via dynamic multi-pooling convolutional neural networks,” 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)*, 2015, pp. 167–176.

[20] Y. Chen, H. Yang, K. Liu, J. Zhao, and Y. Jia, “Collective event detection via a hierarchical and bias tagging networks with gated multi-level attention mechanisms,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 1267–1276.

[21] D. Lou, Z. Liao, S. Deng, N. Zhang, and H. Chen, “Mlbinet: A cross-sentence collective event detection network,” *arXiv preprint arXiv:2105.09458*, 2021.

[22] H. Yan, X. Jin, X. Meng, J. Guo, and X. Cheng, “Event detection with multi-order graph convolution and aggregated attention,” 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)*, 2019, pp. 5766–5770.