Journal of Network Computing and Applications (2025)
Clausius Scientific Press, Canada

DOI: 10.23977/jnca.2025.100106
ISSN 2371-9214 Vol. 10 Num. 1

# *Entity Relation Extraction for Table Filling Based on Dynamic Convolution*

**Jin Chen**

*School of Information, North China University of Technology, Beijing, 100144 China*

***Keywords:*** Joint Relation Extraction, Named Entity Recognition, Dynamic Convolution, Natural Language Processing

***Abstract:*** To address the formidable challenges of complex contextual dependencies and high resource consumption in capturing long-distance relationships in joint entity-relation extraction tasks, a novel and innovative model is proposed. This model strategically leverages dynamic convolution to revolutionize the way the task is approached. Specifically, it reformulates the joint entity-relation extraction task as table annotation, ingeniously treating tables as if they were images, with each cell within the table corresponding to a pixel. By doing so, it creates a unique and structured framework for analysis. Dynamic convolution is then employed in a sophisticated manner to enhance the modeling of local dependencies. This not only allows for a more nuanced understanding of the data but also effectively improves the representation of the intricate and often convoluted relationships between entities. Additionally, the model incorporates an efficient feature extraction strategy. This strategy is carefully designed to significantly reduce computational resource usage, ensuring that the model can operate smoothly without sacrificing performance. To validate the effectiveness of the proposed model, extensive and rigorous experiments are conducted on well-known benchmark datasets, including CoNLL04, ACE05, and ADE. The comprehensive experimental results clearly demonstrate that the model not only improves the accuracy of entity and relation extraction but also achieves the remarkable feat of reducing resource consumption, making it a promising solution in the field.

## 1. Introduction

Obtaining named entities and their relationships from natural language text serves as a crucial initial step in knowledge extraction and knowledge graph construction [1]. Conventional entity-relation extraction methods can generally be classified into pipeline-based and joint extraction approaches. Pipeline methods first carry out named entity recognition [2] and then perform relationship classification [3]. Although these methods are simple and flexible, their separated design often causes problems like insufficient information exchange, error accumulation, and redundant entity identification. On the contrary, joint extraction models, which handle entity recognition and relationship extraction simultaneously, can mitigate these problems to a certain degree.

In recent years, language encoding models grounded in the Transformer framework [4]—such as BERT [5], ERNIE [6], and RoBERTa [7]—Many models have shown great capabilities in natural language processing, especially in joint entity-relation extraction. However, they often require

extremely high computational resources. The Transformer architecture, with its multi-layer attention mechanisms for handling extensive context, results in large-scale model parameters[8-9]. This leads to long training times and a high demand for computational resources like GPUs or TPUs. Although good at capturing long-distance dependencies, it comes at high memory and computational costs. When dealing with large-scale datasets, long training periods and high energy consumption become more serious. So, solving these problems while maintaining performance is a key research goal. Reducing resource consumption without sacrificing performance is a central focus in current research. To address this, this paper presents a joint extraction model using dynamic convolution in a table - filling framework. The main contributions of this study are as follows: A new model design is proposed, integrating table-filling and image-processing concepts, reframing entity-relation extraction as a table annotation task. Dynamic convolution is used to enhance local-dependency modeling for efficient feature extraction. The model performs well on multiple benchmark datasets, significantly reducing model complexity and resource consumption [10-11].

## 2. Relation Work

### 2.1 Dynamic Convolution

Dynamic convolution is an effective feature extraction method that creates real-time weighted combinations of expert convolution kernels based on input features, enhancing adaptability and processing efficiency [16]. It optimizes parameter use, expanding model capacity without significantly increasing FLOPs, making it ideal for low-FLOPs models in large-scale visual pre-training. This technique excels in image classification, object detection, and overcoming low-FLOPs limitations in large-scale pre-training. In our research, dynamic convolution's local-focusing property improves entity-relation extraction by capturing detailed local dependencies, enhancing accuracy in representing complex entity connections.

### 2.2 Joint Entity-Relation Extraction

In the joint entity - relation extraction field, accurately capturing complex context and long-distance relationships demands vast computational resources and memory. Existing methods, like Zheng et al.'s shared encoder method, Hong et al.'s use of GCNs, Zheng and Hao et al.'s semi-supervised learning, and Zeng et al.'s context - aware cross-encoder architecture, have tried to enhance extraction, but they all suffer from high resource consumption. For instance, Zheng et al.'s method still consumes many resources for long-distance dependencies, GCNs bring high computational complexity, semi-supervised learning has complex task design and high resource use, and Zeng et al.'s architecture requires a lot of resources for long - distance dependencies. Thus, to address these issues, this research proposes a table - filling joint extraction model based on dynamic convolution, which can reduce resource consumption while handling complex contexts and long-distance relationships by strengthening local-dependency modeling.

## 3. Proposed Method

### 3.1 Joint Entity-Relation Extraction

Fig. 1 shows the table - filling strategy for an upper triangular matrix Y with dimensions $n \times n$, is equivalent to the length of the input sentence. The diagonal elements $Y_{i, i}$ are associated with the named entity labels of the word $w_i$, and the off - diagonal elements $Y_{i, j}$ are used for relationship annotations. To exclude the lower triangular part of the table during the relationship extraction task,

we adopt the directional encoding method for relationship labels, as proposed by Zhang et al. [12].

Entity Recognition: When it comes to entity recognition, the BILOU tagging system (Begin, Inside, Last, Outside, Unit - length) is utilized to precisely determine the boundaries of named entities. Every word in the sentence is given a corresponding BILOU label. This approach enables the accurate identification of the ranges that entities occupy.

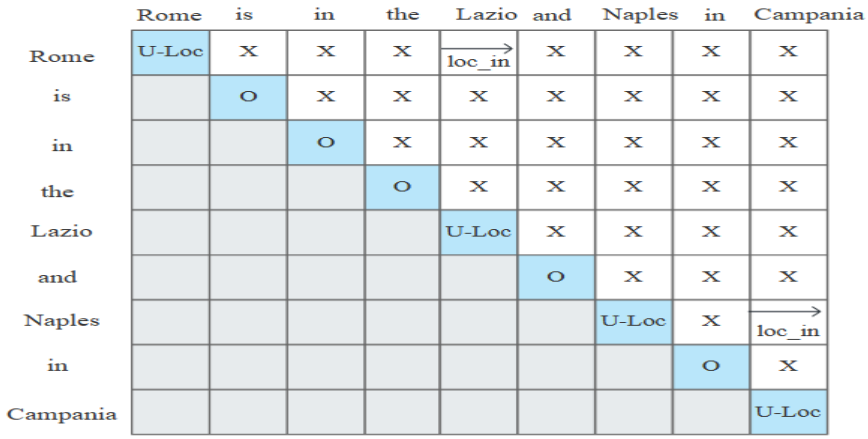


Fig. 1. Example of the table filling strategy

Relation Extraction: the off diagonal positions within the matrix represent the relationship state between entities. Locations with an existing relationship are marked with directional labels like $\overrightarrow{\text{loc\_in}}$,, while positions without a relationship are denoted by "×". This encoding strategy simplifies the structure of the table and simultaneously allows for efficient modeling of entity relationships.

The structure of the proposed model is depicted in Fig. 2, which consists of the following crucial modules: Representation Encoding Layer: At the beginning, the input sentence is encoded by a BERT model. This model generates a sequence representation E for the sentence, which serves as the fundamental input for the subsequent steps aiming to automatically generate table annotations.

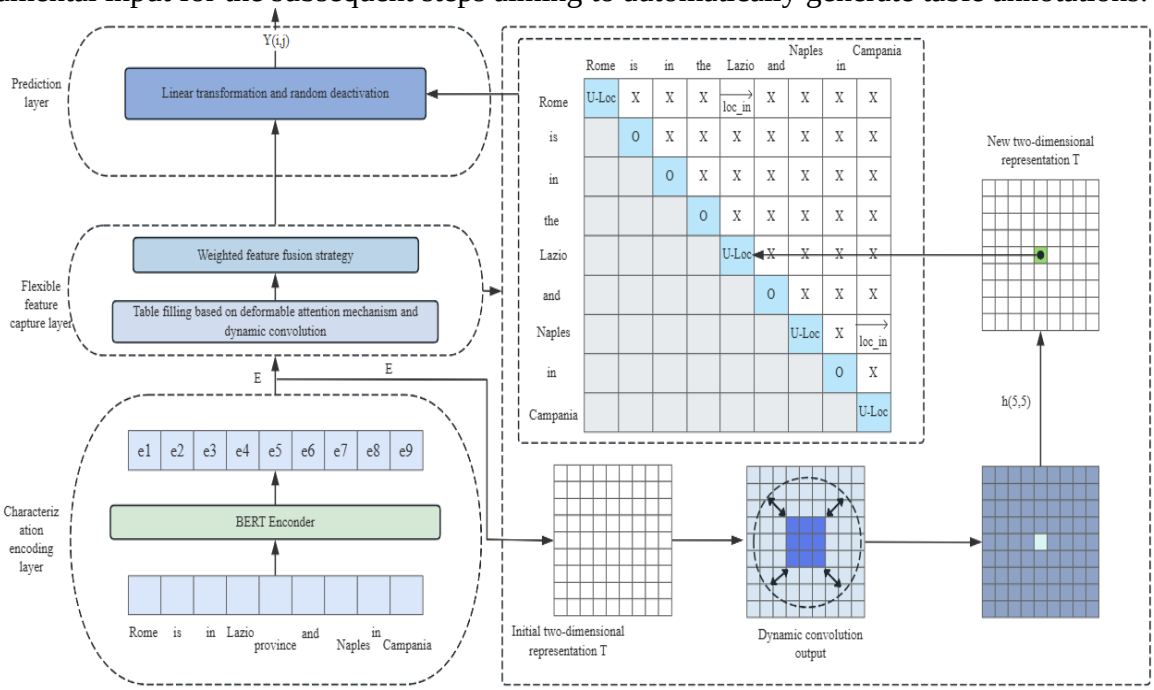


Fig. 2. The model structure diagram

### 3.2 Representation Encoding Layer

The word embeddings are obtained from the contextualized representations produced by the pre - trained BERT model. In line with previous research, the embedding for each word is calculated through max pooling over the subword tokens that make it up. Specifically, for a word $w_i$ consisting of subword tokens, the embedding $e_i$ is computed as follows:

$$e_i := \max\left(x^{(l)}_{start(i)}, \cdots, x^{(l)}_{end(i)}\right) \quad (1)$$

where:

$x^{(l)} \in \mathbb{R}^{d_{emb}}$ represents the output of the pre - trained BERT model.

l is the index of the BERT layer.

$d_{emb}$ is the dimension of the embedding.

max(·) indicates the max - pooling operation carried out across the representations of subword tokens.

### 3.3 Flexible Feature Capture Layer

We draw inspiration from pixel representation in image processing to represent the features of each table cell. From this perspective, we use convolutional operations to effectively capture the local context.

This method strikes a balance between local sensitivity and global dependency modeling, enabling a more accurate feature representation. The self - attention mechanism is mainly used to capture global contextual relationships, and the convolutional operation helps to strengthen the modeling of local dependencies. These two work in synergy to enhance the accuracy and robustness of feature representations. This is particularly beneficial in tasks that demand a deep understanding of context, such as the entity - relation extraction task in this study [13].

#### 3.3.1 Dynamic Convolution

As detailed in reference [16], DynamicConv exhibits a more powerful feature representation ability compared to traditional static convolution. Rooted in the dynamic perceptron concept, the fundamental principle of DynamicConv can be elucidated by the following equations:

$$y = g\left(\bar{W}^T(x)x + \tilde{b}(x)\right) \quad (2)$$

$$\widetilde{W}(x) = \sum_{k=1}^{k} \pi_k(x)\widetilde{W}_k, \tilde{b}(x) = \sum_{k=1}^{K} \pi_k(x)\tilde{b}_k \quad (3)$$

$$\text{s.t.} \quad 0 \le \pi_k(x) \le 1, \sum_{k=1}^{k} \pi_k(x) = 1 \quad (4)$$

In this framework, x represents the input, and y the output. Noticeably , x is involved in two distinct operations. First, it is used to calculate the parameters of the attention mechanism that generates the dynamic convolution kernel. Second, it participates in the convolution operation itself. The symbols W, b, and g represent weights, bias, and the activation function respectively.. $\pi_k(x)$, which is the attention weight of the k-th linear function, is precisely $\bar{W}^T(x)x + \tilde{b}(x)$. It is crucial to remember that this weight changes according to different input values of x.

Dynamic convolution contains K convolutional filters and is designed based on the traditional structure of convolutional neural networks (CNNs)[14]. It also integrates batch normalization (BatchNorm) and the ReLU activation function. The specific architecture of DynamicConv is depicted in Fig. 3.This process generates k attention weights, which are then allocated to the k convolutional kernels of the layer[15].

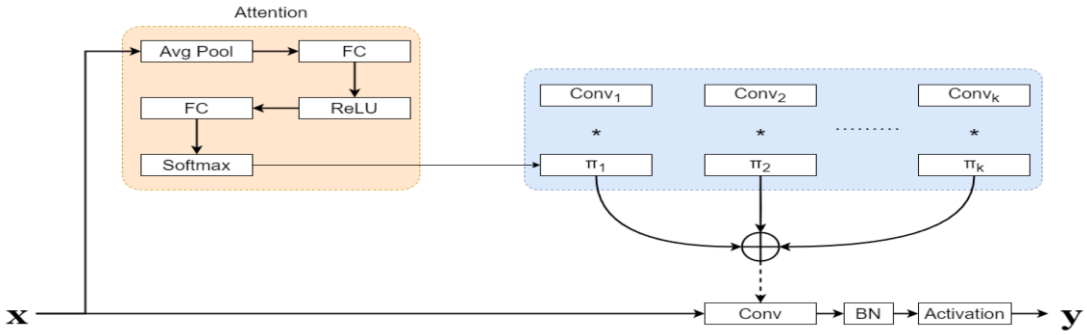


Fig. 3. A dynamic convolution layer

### 3.3.2 Weighted Feature Fusion Strategy

The weighted feature fusion strategy is adjusted to focus on the features from dynamic convolution.

The input x feature undergoes dynamic convolution to obtain the intermediate feature representation $x_{conv1}$, which is then processed with a ReLU activation function and Dropout for non-linearity and regularization:

$$x_{conv1} = \text{Dropout}\left(\text{ReLU}\left(\text{DynamicConv}_1(x)\right)\right) \tag{5}$$

The feature $x_{fused}$ is further processed by another dynamic convolution layer to produce the final output:

$$\text{output} = \text{DynamicConv}_2(x_{conv1}) \tag{6}$$

By this process, the model can better represent local contexts and enhance its performance in complex entity relationship extraction tasks.

## 4. Experiments

This section details the experimental configuration, including the datasets, baseline models, and evaluation approaches..

### 4.1 Experimental Datasets

The proposed system was evaluated on CoNLL04 [17], ACE05, and ADE [18] datasets, with their statistics summarized in Table 1.

For the CoNLL04 and ACE05 datasets, we adhered to the standard evaluation procedures and employed the micro F1 score as the performance indicator. When evaluating on the ADE dataset, the macro F1 score was used. Besides, we measured the model's FLOPs, training duration, and memory usage to evaluate its resource consumption.

Table 1. Statistics of each dataset used in this study

| Dataset | train | dev | test | $\lvert\mathcal{E}\rvert$ | $\lvert\mathcal{R}\rvert$ |
|---|---|---|---|---|---|
| CoNLL04 | 922 | 231 | 288 | 4 | 4 |
| ADE | 4272(10-fold) | | | 2 | 1 |
| ACE05 | 10051 | 2424 | 2050 | 7 | 6 |

### 4.2 Experimental Environment and Training Parameters

The experiments for the model described in this paper were carried out in an environment equipped with Python 3.9, PyTorch 2.3.1, and an M40 - 24G GPU. The specific hyper - parameters of the DYTNET model are presented in Table 2.

Table 2. Hyper- parameters of DYTNET

| Training Config | CONLL04 | ACE05 | ADE |
|---|---|---|---|
| batch size | 8 | 8 | 16 |
| Learning rate (BERT) | 5e-5 | 5e-5 | 5e-5 |
| Learning rate (others) | 1e-3 | 1e-3 | 1e-3 |
| dropout | 0.3 | 0.3 | 0.3 |
| warm-up period | 0.2 | 0.2 | 0.2 |
| epochs | 30 | 30 | 30 |

### 4.3 Results and Discussion

In this part, the performance of the proposed model is assessed by comparing it with several typical baseline methods:

ATG: It is an autoregressive framework that converts text into a graph for the joint extraction of entities and relations [19].

GraphER: This method focuses on graph structure learning for information extraction. It enhances the ability of dynamic optimization and structure - based entity - relation prediction [20].

PFN: A partition filter network designed for joint entity and relation extraction [21].

PURE: A pipeline - based approach for entity and relation extraction [22].

Table-Sequence: An encoder that combines table and sequence encoders. It utilizes their synergy to improve entity recognition and relation extraction [23].

SpERT: A span - based model for joint entity and relation extraction. It boosts performance through lightweight reasoning, entity recognition, and local context - based relation classification [24].

#### 4.3.1 Dataset Accuracy Comparison

Table 3 displays the experimental outcomes of diverse models with encoder pre - trained models of different scales. The performance of relation extraction (RE) is gauged by two metrics: RE and RE+.

RE: A predicted relation is regarded as correct provided that both the relation label and the entity spans are in line with the ground truth.

RE+: For a predicted relation to be deemed accurate, not only the relation label but also the entity labels must be correct.

The symbols △ and ▲ signify evaluations using micro - averaged and macro - averaged F1 scores, respectively.

DYTNet exhibits its edge in Named Entity Recognition (NER) and the more demanding Relation Extraction (RE+) tasks across multiple datasets. For example, on the CoNLL04 dataset, DYTNet attains an NER accuracy of 90.7%, and on the ADE dataset, the NER accuracy hits 90.1%. In the RE+ task, especially on the ADE dataset, DYTNet reaches a performance level of 81.2%, surpassing numerous other models. These results underscore DYTNet's robust ability to efficiently handle intricate relation extraction tasks.

Table 3. Experimental Result

| Datasets | Model | Encoder | NER | RE | RE+ |
|---|---|---|---|---|---|
| CoNLL04△ | SpERT | BERT | 88.9 | - | 71.5 |
| | ATG | BERT | 90.5 | 73.5 | 73.5 |
| | GraphER | BERT | 90.2 | 76.6 | **76.6** |
| | DYTNet(Ours) | BERT | **90.7** | **76.8** | 76.2 |
| ADE▲ | SpERT | BERT | 89.3 | - | 79.2 |
| | PFN | BERT | 89.6 | - | 80.0 |
| | Table-Sequence | ALBERT | 89.7 | 80.1 | 80.1 |
| | DYTNet(Ours) | BERT | **90.1** | **83.3** | **81.2** |
| ACE05△ | GraphER | BERT | 89.8 | 68.4 | 66.4 |
| | Table-Sequence | ALBERT | 89.5 | 67.6 | 64.3 |
| | ATG | BERT | **90.1** | 68.7 | 66.2 |
| | PURE | BERT | 89.7 | 69.0 | 65.6 |
| | DYTNet(Ours) | BERT | 89.8 | **69.7** | **67.2** |

### 4.3.2 Resource Consumption Comparison

As presented in Table 3, the experiments mainly make use of a twelve - layer BERT as the encoder, in which DYTNet shows outstanding performance. Nevertheless, using a twelve - layer BERT substantially raises resource consumption. The details of resource utilization, such as FLOPs, training duration, and memory utilization, are compiled in Table 4. This reveals the compromise between model performance and computational efficiency when larger encoders are employed.

Table 4. Experimental Result

| Datasets | Model | FLOPs | Training time | Memory usage |
|---|---|---|---|---|
| CoNLL04 | SpERT | 22.50G | 1.51h | 70.3 |
| | ATG | 23.42G | 1.47h | 71.5 |
| | GraphER | 22.47G | 1.36h | 74.6 |
| | DYTNet(Ours) | **27.28G** | **1.89h** | **79.7** |
| ADE | SpERT | 23.48G | 0.51h | 75.2 |
| | PFN | 24.39G | 0.67h | 67.5 |
| | Table-Sequence | 26.77G | 0.47h | 49.5 |
| | DYTNet(Ours) | **28.48G** | **0.82h** | **83.3** |
| ACE05 | GraphER | 25.43G | 2.43h | 68.4 |
| | Table-Sequence | 27.41G | 2.03h | 51.2 |
| | ATG | 26.51G | 2.21h | 68.7 |
| | PURE | 23.64G | 2.81h | 69.0 |
| | DYTNet(Ours) | **30.67G** | **3.29h** | **86.4** |

As indicated in the table, when BERT adopts a twelve - layer architecture, the DYTNet model exhibits notably higher resource consumption across the three datasets in comparison to other baseline models. This augmented resource utilization is due to the implementation of dynamic convolution on the outputs of the BERT encoder, which escalates resource requirements. To alleviate this burden, we chose to test the model's performance with a decreased number of BERT layers. Our intention was to assess the influence of encoder depth on the model's efficacy. The figure beneath shows the experimental results of DYTNet during training on the CoNLL04 dataset, with different numbers of BERT layers.

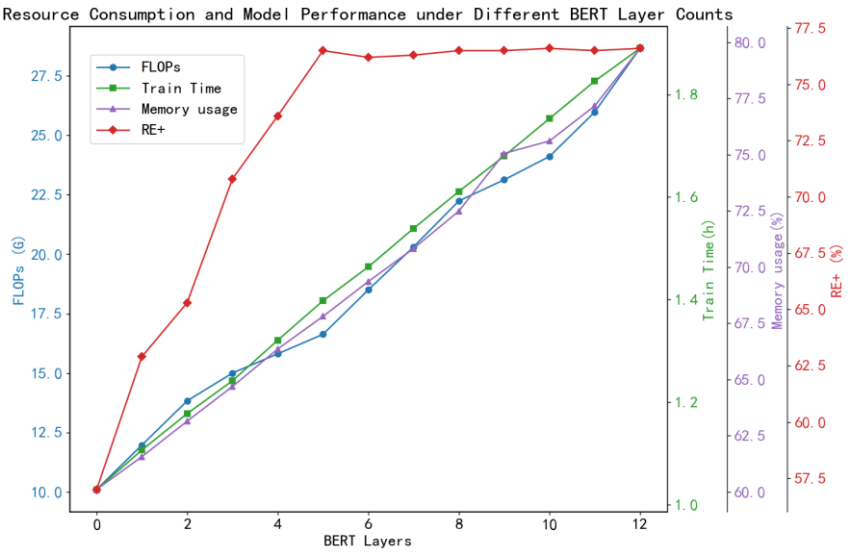


Fig. 4. Training Results.

As depicted in Fig. 4, as the number of BERT layers increases from 0 to 12, the model's FLOPs, training duration, and memory consumption show a linear upward trend. Yet, when the number of BERT layers reaches 5, the RE+ score reaches its maximum value. After this point, additional increases in the number of BERT layers result in only marginal improvements to the RE+ score. In this configuration, DYTNet has a FLOPs value of 21.63G, a training time of 1.398 hours, a memory usage rate of 67.8%, and an RE+ score of 76.8.

This finding implies that DYTNet can achieve the best performance with a relatively less deep BERT encoder. To verify this assumption, we carried out comparative experiments among DYTNet, SpERT, and GraphER, and the experimental results are illustrated in Fig. 5.

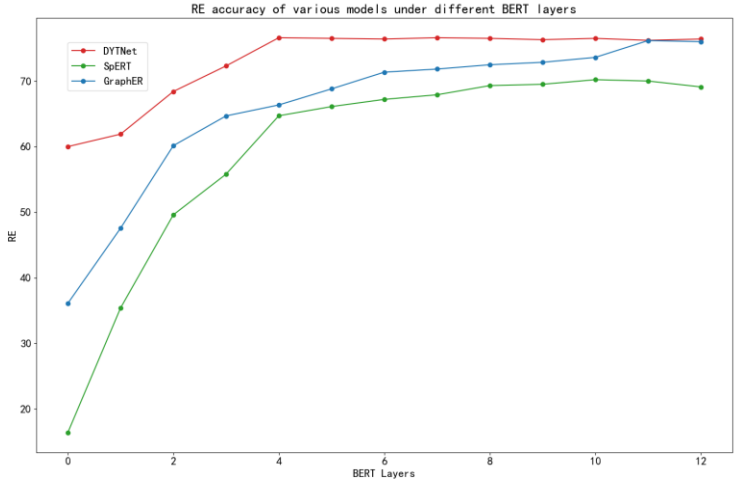


Fig. 5. RE (F1) results of various models on the CoNLL04 dataset

Fig. 5 shows RE F1 score rises linearly for SpERT and GraphER with more BERT layers, while DYTNet peaks at 4 layers, saving resources with high accuracy. Experiments reveal DYTNet captures contextual dependencies well even with fewer layers, balancing performance and efficiency, making it ideal for resource-constrained text relation extraction tasks. We analyzed a 5-layer DYTNet on CoNLL04 dataset for accuracy and resources, outperforming others under optimal conditions (Table 5).

Table 5. Final Experimental Result

| Model | FLOPs | Training Time | Memory usage | NER | RE | RE+ |
|---|---|---|---|---|---|---|
| SpERT | 22.50G | 1.51h | 70.3 | 88.9 | - | 71.5 |
| ATG | 23.42G | 1.47h | 71.5 | **90.5** | 73.5 | 73.5 |
| GraphER | 21.47G | 1.36h | 74.6 | 89.0 | 76.3 | 76.1 |
| DYTNet(Ours) | **16.63G** | **1.31h** | **66.4** | 89.3 | **76.8** | **76.8** |

DYTNet manages to cut down on resource utilization while still upholding high-level performance. When employing 3 to 5 BERT layers, it attains peak or nearly - peak outcomes in both NER and RE tasks. In terms of efficiency, it outperforms SpERT and GraphER, which makes DYTNet an excellent choice for resource - restricted large - scale relation extraction undertakings.

## 5. Conclusion

The proposed DYTNet model innovatively combines table - filling techniques with methods inspired by image processing. This offers a novel solution for the joint entity - relation extraction task. By framing the relation extraction task as a table - filling problem and making use of dynamic convolution, the model efficiently captures local dependencies. This not only boosts extraction accuracy but also decreases model complexity and computational resource consumption. Experimental results indicate that the model performs remarkably well on datasets like CoNLL04, ACE05, and ADE. It has higher extraction accuracy compared to existing methods, all while reducing computational resource demands. Looking ahead, future research directions involve further optimizing the model's architecture to further lower computational complexity. Additionally, efforts will be made to expand the model's applicability to cross - domain tasks. There will also be an emphasis on enhancing generalization and adaptability in low - resource scenarios through transfer learning techniques.

## References


*[1] Bosselut A, Rashkin H, Sap M, et al. COMET: Commonsense transformers for automatic knowledge graph construction[J]. arXiv preprint arXiv:1906.05317, 2019.*

*[2] Jehangir B, Radhakrishnan S, Agarwal R. A survey on Named Entity Recognition—datasets, tools, and methodologies[J]. Natural Language Processing Journal, 2023, 3: 100017.*

*[3] Zhao X, Deng Y, Yang M, et al. A comprehensive survey on relation extraction: Recent advances and new frontiers[J]. ACM Computing Surveys, 2024, 56(11): 1-39.*

*[4] Ashish V. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30: 1.*

*[5] Kenton J D M W C, Toutanova L K. Bert: Pre-training of deep bidirectional transformers for language understanding[C]//Proceedings of naacL-HLT. 2019, 1: 2.*

*[6] Sun Y, Wang S, Li Y, et al. Ernie: Enhanced representation through knowledge integration[J]. arXiv preprint arXiv:1904.09223, 2019.*

*[7] Liu Y. Roberta: A robustly optimized bert pretraining approach[J].arxiv preprint arxiv:1907.11692, 2019.*

*[8] Zheng Y, Tuan L A. A novel, cognitively inspired, unified graph-based multi-task framework for information extraction[J]. Cognitive Computation, 2023, 15(6): 2004-2013.*

*[9] Hong Y, Liu Y, Yang S, et al. Improving graph convolutional networks based on relation-aware attention for end-to-end relation extraction[J]. IEEE Access, 2020, 8: 51315-51323.*

*[10] Zheng Y, Hao A, Luu A T. Jointprop: joint semi-supervised learning for entity and relation extraction with heterogeneous graph-based propagation[J]. arXiv preprint arXiv:2305.15872, 2023.*

*[11] Zeng D, Xu L, Jiang C, et al. Sequence tagging with a rethinking structure for joint entity and relation extraction[J]. International Journal of Machine Learning and Cybernetics, 2024, 15(2): 519-531.*

*[12] Ma Y, Hiraoka T, Okazaki N. Named entity recognition and relation extraction using enhanced table filling by contextualized representations[J]. Journal of Natural Language Processing, 2022, 29(1): 187-223.*

*[13] Zhang M, Zhang Y, Fu G. End-to-end neural relation extraction with global optimization[C]//Proceedings of the*

*2017 conference on empirical methods in natural language processing. 2017: 1730-1740.*
*[14] Dosovitskiy A. An image is worth 16x16 words: Transformers for image recognition at scale[J]. arXiv preprint arXiv:2010.11929, 2020.*
*[15] Xia Z, Pan X, Song S, et al. Vision transformer with deformable attention[C]. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022: 4794-4803.*
*[16] Han K, Wang Y, Guo J, et al. ParameterNet: Parameters Are All You Need for Large-scale Visual Pretraining of Mobile Networks[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2024: 15751-15761.*
*[17] Roth D, Yih W. A linear programming formulation for global inference in natural language tasks[C]. Proceedings of the eighth conference on computational natural language learning (CoNLL-2004) at HLT-NAACL 2004. 2004: 1-8. Available: https://aclanthology.org/W04-2401*
*[18] Gurulingappa H, Mateen-Rajpu A, Toldo L. Extraction of potential adverse drug events from medical case reports[J]. Journal of biomedical semantics, 2012, 3: 1-10.*
*[19] Zaratiana U, Tomeh N, Holat P, et al. An Autoregressive Text-to-Graph Framework for Joint Entity and Relation Extraction[C]. Proceedings of the AAAI Conference on Artificial Intelligence. 2024, 38(17): 19477-19487.*
*[20] Zaratiana U, Tomeh N, Khbir N E, et al. GraphER: A Structure-aware Text-to-Graph Model for Entity and Relation Extraction[J]. arXiv preprint arXiv:2404.12491, 2024.*
*[21] Yan Z, Zhang C, Fu J, et al. A partition filter network for joint entity and relation extraction[J]. arXiv preprint arXiv:2108.12202, 2021.*
*[22] Zhong Z, Chen D. A frustratingly easy approach for entity and relation extraction[J]. arXiv preprint arXiv:2010.12812, 2020.*
*[23] Wang J, Lu W. Two are better than one: Joint entity and relation extraction with table-sequence encoders[J]. arXiv preprint arXiv:2010.03851, 2020.*
*[24] Eberts M, Ulges A. Span-based joint entity and relation extraction with transformer pre-training[M]. ECAI 2020. IOS Press, 2020: 2006-2013.*