LAI: Label Annotation Interaction based Representation Enhancement for End to End Relation Extraction

Rongxuan Lai , Wenhui Wu , Li Zou , Feifan Liao , Zhenyi Wang * , Haibo Mi *

Posted Date: 26 May 2025

doi: 10.20944/preprints202505.1990.v1

Keywords: Nature Language Process; Jointly Entity and Relation Extraction; Information Extraction; Graph Neural Network; Representation Enhancement

 doi:10.20944/preprints202505.1990.v1

Article

LAI: Label Annotation Interaction based Representation Enhancement for End to End Relation Extraction

Rongxuan Lai [1,2],Wenhui Wu [1], Li Zou [1,2], Feifan Liao [1,2], Zhenyi Wang [1,2,*,†], Haibo Mi [1,2,*,†]

[1] College of Information and Communication, National Defense Technology University, Wuhan 430001, China
[2] State Key Laboratory of Complex & Critical Software Environment, Wuhan 430001, China
* Correspondence: wzy3503@163.com (Z.W.); haibo_mihb@nudt.edu.cn (H.M.)

Abstract: End-to-end relation extraction (E2ERE) generally performs named entity recognition and relation extraction either simultaneously or sequentially. While numerous studies on E2ERE have centered on enhancing span representations to improve model performance, challenges remain due to the gaps between subtasks (named entity recognition and relation extraction) and the modeling discrepancies between entities and relations. In this paper, we propose a novel **L**abel **A**nnotation **I**nteraction based representation enhancement method for E2ERE, which institutes a two-phase semantic interaction to augment representations. Specifically, we firstly feed label annotations that are easy to manually annotate into a language model, and conduct the **first round interaction** between three types of tokens with a partial attention mechanism; Then construct a latent multi-view graph to capture various possible links between label and entity (pair) nodes, facilitating the **second round interaction** between entities and labels. A series of comparative experiments with methods of various transformer-based architectures currently in use have shown that LAI-Net can maintain performance on par with the current SOTA in terms of NER task, and achieves significant improvements over existing SOTA models in terms of RE task.

Keywords: nature language process; jointly entity and relation extraction; information extraction; graph neural network; representation enhancement

1. Introduction

As a core information extraction (IE) task, end-to-end relation extraction (E2ERE) can be split into named entity recognition (NER) subtask for entity identification and relation extraction (RE) subtask for capturing inter-entity relations from plain texts. As postulated by [**?**], E2ERE is challenging for its difficulty in capturing affluent correlations between entities and relations. IE research has traditionally converted NER and RE tasks into span-based tasks [**? ? ? ? ?**]. Though these methodologies have incrementally advanced model performance from various perspectives, they are still impeded by two pivotal limitations: overdetached of sub-tasks leads to insufficient information exchange between entitiy and relation, and the disparity in modeling strategies between entity and relation result in semantic gaps. In this paper, we mainly focus on enhancing semantic interaction during modeling process to achieve enhanced span representation for E2ERE.

To address the challenges above, prevailing researches mainly focuses on reorganizing the input or intermediate network layers of pre-train language model (PLM), attempting to enhance the semantic information of representation through the integration of specialized symbols or extrinsic prior knowledge. We roughly divided into three types (as shown in Figure 1): **Vanilla based** method is a straightforward approach to acquiring a given span representation by feeding raw text tokens series into pre-trained encoder. **Marker based** method inserts independent entity markers like [M], [\M] amidst text tokens to highlight the presence of entities, aims at attracting more model attention. **Enumerate based** method enumerates all posible entity candidates from plain text and then concatenate

 doi:10.20944/preprints202505.1990.v1

them after text tokens, entity tokens share the position ids with candidates as well. For these methods above, the distinguished LSTM-CRF [**?**] is a typical vanilla based sequence labeling method, and PURE [**?**] is a combination of marker based method and enumeration based method, which adopts marker based method during NER phase and enumerate based method for RE phase, that achieved SOTA performance. PL-Marker [**?**] is a typical enumeration based method that promoted the SOTA further.

In addition to the above three methods, what we develop in this paper can be classified as the fourth class named **annotate & enumeration based** method. It's an novel semantic enhancement approach with external knowledge, inspired by external knowledge based aproaches [**? ? ?**]. We argue that a thorough comprehension of label semantics will significantly enhance the IE model abilities, what serves as a premise for our work.

Figure 1. An illustration of different representation enhancement methods. t indicates an individual token from text. The bordered rectangles highlighted in assorted colors signify discrete elements: blue, light yellow (or green), pink respectively indicate entities markers, groups of entity tokens and trailed markers, groups of label annotation tokens. Bidirectionally connected squares sharing the same color refer to elements that have identical position id.

As shown in Figure.1, our principal improvement over preceding methods lies in the insertion of external prior knowledge (i.e., label annotations) into the PLM input sequence, aiming to leverage PLM's internal layers to enhance semantic interaction between label and text. This represents the first round semantic interaction in LAI-Net framework.

Unlike the lexicon adapter-based methods [**? ?**] and lattice-based methods [**? ? ?**], we manually expand the label information and embed it between text tokens and enumerated candidates then feed the series into a pre-trained model. We further enhance the representation by combining word vectors using downstream neural networks. Formally, the augmented representations derived from the aforementioned methods can be summarized as:

$$\mathbf{h}_{\text{span}}^{V} = f\left(\mathbf{h}_{\text{span}}^{s}; \mathbf{h}_{\text{span}}^{e}\right) \tag{1}$$

$$\mathbf{h}_{\text{span}}^{M} = f\left(\mathbf{h}_{M}; \mathbf{h}_{\setminus M}\right) \tag{2}$$

$$\mathbf{h}_{\text{span}}^{E} = f\left(\mathbf{h}_{\text{span}}^{s}; \mathbf{h}_{\text{span}}^{e}; \mathbf{h}_{M}; \mathbf{h}_{\setminus M}\right) \tag{3}$$

$$\mathbf{h}_{\text{span}}^{A\&E} = f\left(\mathbf{h}_{\text{span}}^{s}; \mathbf{h}_{\text{span}}^{e}; \mathbf{h}_{M}; \mathbf{h}_{\setminus M}; \mathbf{h}_{a}^{s}; \mathbf{h}_{a}^{e}\right) \tag{4}$$

where $f(\cdot)$ is an tensor operator, equipped to execute series of operations including tensor addition, tensor multiplication, tensor concatenation, etc., or even could be a neural network. And superscripts s, e denote the commencement and termination tokens of a span or annotation, while the subscripts span, M, $\setminus M$ represent the disparate token types.

Based on the first round semantic interaction, we further meticulously crafted with selectively designed downstream network layers. The main innovations and improvements lie in the following aspects:

- We explore a novel two-round semantic interaction approach for enhancing span representations, wherein the first round interaction reorganizes the PLM input with annotated information in what we term an annotation & enumeration-based method, and the second round interaction employs GCN built atop Gaussian Graph Generator modules to facilitate label semantic fusion.
- We conduct a coarse screening with a entity candidate filter to eliminate out spans that are clearly not real entities, which also promotes the saving of computing resources.
- Experiments demonstrates that our method, while slightly lagging behind current SOTA in NER performance, takes the lead in the downstream RE task, surpassing the current SOTA performance.

 doi:10.20944/preprints202505.1990.v1

2. Related Work

Recently, academic interest in span representation enhancement has surged, providing a substantial impetus to E2ERE. Traditional neural network based methods often ignore non-local and non-sequential context information from input text [**?**], what is exactly GCNs [**? ?**] excel in. GCNs, what our discussion centered on, have been widely used to model the interaction between entities and relations in the text, and has been demonstrated as a typical and effective approach. GCN-based approaches typically leverage a predefined graph structure, which constructed from plain text, to facilitate information propagation among nodes, thus capturing text's non-linear structure and enhancing NER and RE models' capabilities to capture both global graph structures and representations of nodes and edges. Mostly GCN based methods [**? ? ? ? ? ?**] utilizes different approaches to define nodes (sentences, words, tokens, spans, labels, etc.), and these nodes can be connected by syntactic dependency edges [**? ? ? ?**], re-occurrence edges[**?**], co-occurrence edges[**? ?**], adjacent word edges[**? ? ?**], adjacent sentence edge[**?**], etc. And then perform convolution operations on the graph to facilitate the flow of information between nodes, which enables nodes to efficiently acquire both local and global information. This further refines node representations and downstream network performance. Building on these advances, our work also adopt a GCN based method to design the interaction between entities and relations.

3. Methodology

3.1. Task Definition

Given a sentence formularized as $S = \{w_1, w_2, \cdots, w_m\} = \{t_1, t_2, \cdots, t_n\}$ with m words (or n tokens, $n \geq m$ naturally), the goal of E2ERE task is to recognize a set of entity spans and relationships of entity pairs automatically, which can be written as $(e_i, r_{i,j}, e_j) \in \mathcal{T}$. Every entity e, which attaches specific type (e.g. person (`PER`), organization (`ORG`)), is a sequence of tokens. Every relation $r_{i,j}$ represents the relationship between e_i and e_j, and also attaches a specific type (e.g. organization affiliation relation (`ORG-AFF`))[1]. Formularly, we define the set of possible entity types and relation types as $\mathcal{E}$, $\mathcal{R}$ respectively.

3.2. First Round Semantic Interaction

During data processing, we concatenate text tokens, annotation tokens, and marker tokens sequentially to formulate a unified input sequence (the bottom element of Figure 2). The PLM encoder conducts the first round semantic interaction, then output the encoded representation, which is semantic amalgamation of the three types tokens.

Text Tokens Our approach breaks down the words from raw text into text token sequences as part of the model input.

Annotation Tokens Inspired by [**?**] and [**?**], we augment semantic information by manually annotating the entity (or relation) abbreviated label both in NER and RE phase. For example, the abbreviated entity type `GPE` can be annotated as "geography political entity", a fully-semantic unbroken phrase. Correspondingly, the abbreviated relation type `ORG-AFF` can be annotated as "organization affiliation". Each label is manually expanded to enrich semantic content and then tokenized into annotation tokens (highlighted by red rectangle in bottom of Figure 2), which are appended to the text tokens sequentially.

Marker Tokens We enumerate all potential consecutive token sequences (i.e. entity candidates) not exceeding a predefined limitation of length c (with $c \leq n$) within a sentence, labeling each with an entity type. If $c = 2$, as shown in Figure 2, the set of all the possible spans from sentence "`chalabi is the founder and leader of the iraqi national congress.`" can be written as $\mathbf{\Psi} = \{$"chalabi", "chalabi is", "is", "is the", "the founder", "founder", "founder and", "and", "and leader"$\cdots\}$. The i-th span can be written as $\text{span}_i = [\text{span}_i^s, \text{span}_i^e]$, where span_i^s and span_e^e are indicative of start and

[1] Both entity type sample and relation type sample mentioned above are quoted from ACE05.

 doi:10.20944/preprints202505.1990.v1

end position id of entity span respectively. Therefore, entity candidate series can also be written as $\mathbf{\Psi} = \{[0,0],[0,1],[1,1],[1,2],[2,2],[2,3],[3,3],[3,4],[4,4],[4,5]\}$ given position id perspective. Thus, the number of candidate for a sentence with m words:$|\mathbf{\Psi}| = m \cdot c + (c - c^2)/2$.

In model input, we define a start marker (M) and an end marker ($\backslash M$), which form a pair of marker tokens, respectively represent the start and end of an entity span and are appended subsequent to annotation tokens. The start and end marker share the same position embedding with corresponding span's start token and end token respectively, while keeping the position id of original text tokens unchanged. From an PLM encoder input perspective, every marker is a token element of tokens series, called marker token. As shown in Figure 2, entity `chalabi` is highlighted by light yellow bordered square, and its corresponding markers noted by a colorful non-bordered square with line frame differred in various entity labels (white means non-entity). In conclusion, the complete input sequence can be represented as Eq.(5), where a_i is a token broken from label annotation, $\mathrm{M}_i^s, \mathrm{M}_i^e$ represent start and end marker token respectively.

$$\widetilde{S} = \{[\mathrm{CLS}], t_0, \cdots, t_{n-1}, [\mathrm{SEP}], a_0, \cdots, a_{N-1}, [\mathrm{SEP}], \mathrm{M}_0^s, \cdots, \mathrm{M}_{|\mathbf{\Psi}|-1}^s, \mathrm{M}_0^e, \cdots, \mathrm{M}_{|\mathbf{\Psi}|-1}^e\} \tag{5}$$

Partial Attention Although special tokens, such as `[CLS]` and `[SEP]` serve to isolate different types of tokens, there still exists semantic interference among them. The straightforward blend of annotation tokens and marker tokens with text tokens may disrupt semantic consistency of raw text. To mitigate this, we devise a partial attention mechanism, allowing selective semantic influence among the differ types of token. This mechanism can effectively control the information flow (could be regard as a kind of visibility) between different tokens, by adjusting the value of elements of the attention mechanism mask matrix. It suppresses the information interaction among tokens mutual invisible, while enhancing the information interaction among tokens mutual visible. Experimental results show that partial attention effectively improves model performance. See Appendix B for more detail information about partial attention.

Figure 2. Illustration of LAI-Net, where annotatation interaction is highlighted by red color. The left and right part of the architecture represent the RE and NER phase respectively. In addition, ▲ indicates the span representation and differed type tokens marked with same color with their corresponding embeddings.

3.3. Second Round Semantic Interaction

To refine semantic integration, we introduce second round semantic interaction, employing a semantic integrator that explicitly model interactions between entity candidates and label annotations. The semantic integrator consists of multiple GCN layers with randomly generated adjacency matrix, treats both entity spans and label annotations as nodes, and establishs connections between nodes through the construction of a graph $\mathcal{G}$. So that the interactions between nodes can be explicitly modeled. A GCN typically necessitates a manually predefined and fixed adjacency matrix to depict the inter-nodes connections. The fixed adjacency matrix fixes the perspective from which the model understands the semantics. However, it's naturally to note that the inter-nodes connection cannot be predetermined accurately when considering our task. Otherwise, our task would be meaningless. Therefore, inspired by [**?**] and [**?**], we forgo a static adjacency matrix in favor of a multi-view graph, called **G**aussian **G**raph **G**enerator based **G**raph **C**onvolutional **N**etworks (G^{4}CN). G^{4}CN unlike vanilla GCN which takes a fixed adjacency matrix, it gives up fixing the edge weights and set the edge weights via trainable neural networks during the network initial stage, which allows model to assimilate semantic contexts from multiple perspectives.

First, we attach every node with a Gaussian distribution $\mathcal{N}(\mu, \sigma)$, where both μ, σ are generated by trainable neural networks, formulated as in Eq.(6), (7), and set the activation function ϕ as the SoftPlus function, since the standard deviation of Gaussian distribution is bounded on $(0, +\infty)$. Then, simulate edge weight through coumpute the KL divergence w_{ij}^e between two Gaussian distribution of node,

 doi:10.20944/preprints202505.1990.v1

where $w_{ij}^{e} = \mathrm{KL}(\mathcal{N}(\mu_i, \sigma_i) \| \mathcal{N}(\mu_j, \sigma_j))$. So we can obtain a number of Gaussian distributions for the multi-view graph, each $\mathcal{N}_i$ here corresponds to a node representation v_i. Thus, a vanilla GCN's value-fixed adjacency matrix $\mathbf{A} = (a_{ij}^{e})_{k\times k}$ can be modified to be a G^{4}CN's value-varied adjacency matrix $\mathbf{A}^{'} = (w_{ij}^{e})_{k\times k}$, where k is number of nodes. The whole process can be formulated as Eq.(8), where $\mathbf{H}_{\text{span}}, \mathbf{H}_{\text{anno}}^{\text{ent}}$ are matrixs formed by concatenating multiple nodes (span or annotation) in constructed heterogeneous graph $\mathcal{G}$, and $\mathbf{GCN}(\cdot)$ is vanilla GCN, see detail formulas in Appendix A.

$$\{\mu_i^1, \mu_i^2, \mu_i^3, \cdots, \mu_i^N\} = g_\theta(v_i) \tag{6}$$

$$\{\sigma_i^1, \sigma_i^2, \sigma_i^3, \cdots, \sigma_i^N\} = \phi(g_\theta^{'}(v_i)) \tag{7}$$

$$\widetilde{\mathbf{H}}_{\text{span}} = \frac{1}{2}\Big(\mathbf{H}_{\text{span}} + \mathbf{GCN}\Big(\mathbf{A}_{\text{ner}}^{'}, [\mathbf{H}_{\text{span}}; \mathbf{H}_{\text{anno}}^{\text{ent}}]\Big)\Big) \tag{8}$$

3.4. Name Entity Recognition

Span and Annotation Representation We extract the contextualized representations $\mathbf{h}$ for individual token s from PLM output, and naturally obtain the involved mathematical formulas for spans and annotatations as Eq.(9) and Eq.(10), where $\mathbf{h}_{\text{anno}} \in \mathbb{R}^d, \mathbf{h}_{\text{span}} \in \mathbb{R}^d$. And $\mathbf{h}_a^s, \mathbf{h}_a^e$ is embedding of first and last token of a certain type of label annotatation, respectively. $\mathbf{h}_t^s, \mathbf{h}_t^e$ is embedding of first token and last token of a entity candidate respectively, and $\mathbf{h}_M^s, \mathbf{h}_M^e$ indicates the embedding of start and end token of marker respectively. Linear layer **FC** used to harmonize dimensional space.

$$\mathbf{h}_{\text{anno}} = \mathbf{FC}_a([\mathbf{h}_a^s; \mathbf{h}_a^e]) \tag{9}$$

$$\mathbf{h}_{\text{span}} = \mathbf{FC}_{\text{span}}\Big([\mathbf{h}_{\text{span}}^s; \mathbf{h}_{\text{span}}^e; \mathbf{h}_M^s; \mathbf{h}_M^e]\Big) \tag{10}$$

Entity Candidates Filter Before the development of the entity candidates filter module, we attempted to train the model, but it ended in failure. The reasons can be summarized as: 1) During the initialization phase of training, model parameters are randomly assigned values, leading to lots of candidate entities being randomly predicted as real entities in early phase. This further causes the adjacency matrix of the graph neural network to become excessively large in scale, resulting in extremely slow network training and significant resource consumption. 2) Among the numerous candidate entities enumerated, positive sample entities are extremely few while negative sample entities are abundant. This easily induces the model to tend towards classifying all entities as non-entities to achieve the minimum loss value, which is not the desired outcome. To overcome the aforementioned issues, we devise a binary classifier acts as a entity filter, performing coarse screening for all enumerated entities by discarding non-genuine entities, thus optimizing subsequent predictions.

As for loss function, the primary aim of entity filter is to maximize the likelihood function, what drove us follow [**?**] to adopt likelihood loss function as Eq.(11), where $\mathbf{\Psi}_g \subseteq \mathbf{\Psi}$ indicates a set of real entity spans. In addition to intuitive time consumption optimization, experimental results indicate that entity filter successfully alleviates model weakening engendered by negative samples and enhances overall performance.

Span Classifier We conduct span representations classification through a linear classifier, utilizing cross-entropy loss to direct the learning process. The combined loss function $\mathcal{L}_{\text{ner}} = \mathcal{L}_{\text{filter}} + \mathcal{L}_{\text{span}}$ is optimized during training, using $\mathcal{L}_{\text{span}}$ from Eq.(11)-(12), with dropout layers for regularization.

In addition, [**?**] had proved that packing a series of related spans into a training instance can promote the NER model performance, that naturally prompt us to follow the effective measures when reorganize input.

$$\mathcal{L}_{\text{filter}} = -\frac{1}{|\mathbf{\Psi}|}\sum_{i=1}^{|\mathbf{\Psi}|} \log P(\text{span}_i \in \mathbf{\Psi}_g | \text{span}_i \in \mathbf{\Psi}) \tag{11}$$

$$\mathcal{L}_{\text{span}} = -\frac{1}{|\mathbf{\Psi}_g|}\sum\nolimits_{\mathbf{\Psi}_g} \log P_{\text{span}} \tag{12}$$

 doi:10.20944/preprints202505.1990.v1

3.5. Relation Extraction

Subject marker In RE phase, we design our Annotate & Enumeration Based method (as demonstrated in Figure 1) to acquire the enhanced representation. Concretely, we adopt the marker based approach (shown in Figure 1), and insert a pair of subject markers, called solid markers by [**?**], into left and right of subject entity, and enumerated object candidate spans following on the heels of annotatation tokens to extract relations involving the subject entity.

Entity pairs Representation We match subject and object representations up pairwise to obtain a series of entity pairs ($\mathbf{h}_{\text{pair}} = [\mathbf{h}_{\text{subj}}; \mathbf{h}_{\text{obj}}]$). And the label semantic confused pair representation formulas can be written as Eq.(13), where $\mathbf{H}_{\text{pair}}, \mathbf{H}^{\text{rel}}_{\text{anno}}$ are matrixs formed by concatenating entity pair or relation label annotation representations.

$$\widetilde{\mathbf{H}}_{\text{pair}} = \frac{1}{2}\left(\mathbf{H}_{\text{pair}} + \mathbf{GCN}\left(\mathbf{A}'_{\text{re}}, \left[\mathbf{H}_{\text{pair}}; \mathbf{H}^{\text{rel}}_{\text{anno}}\right]\right)\right) \tag{13}$$

The loss function of RE phase is the cross-entropy loss.

4. Experiments

Datasets We utilize 3 widely used standard corpora: 1) ACE05 spans various domains (newswire, online forums), and contains 7 entity types and 6 relation types between entities. 2) SciERC [**?**] is a scientific dataset built from AI conference/workshop proceedings across four communities. It includes 7 entity types and 7 relation types. 3) ADE [**?**] consists of 4,272 sentences and 6,821 relations extracted from medical reports.

Metrics The model with best F1 performance on test set will be selected on a fixed number of epochs. Both micro and macro average metrics are used to evaluate the model performance, former for ACE05/SciERC and latter for ADE. For NER task, an entity prediction is correct if and only if its type and boundaries both match with those of a gold entity. For RE task, a relation prediction is considered correct if its relation type and the boundaries of the two entities match with those in the gold data. We also report the strict relation F1 (denoted RE+), where a relation prediction is considered correct if its relation type as well as the boundaries and types of the two entities all match with those in the gold data. We show detailed experimental settings in Appendix B.

4.1. Main Results

To ensure the credibility of experimental results, we conducted multiple experiments with different random seeds on all datasets (relevant settings are detailed in Appendix B.1), and the stability of running are presented in Appendix B.10.

4.1.1. Results Against Horizontal Comparison

Table 1 presents a horizontal comparison with baselines, focusing on comparing excellent models developed in recent years, including several previous SOTA models (detail in Appendix B.2). In terms of NER task, our method is on par with the current SOTA, with the discrepancy across three datasets ranging from 0.39% to 0.1%, which achieves sub-optimal performance. Starting from a slightly lower NER performance compared to the SOTA, our method still achieved a performance gain of 2-10% on relation F1 and strict relation F1, consistently outperforming all selected baselines, even with the error propagation between NER and RE task. For example, on ACE05, our method exhibits a 0.24% disadvantage in NER phase, but takes the lead in the downstream RE phase, surpassing the current SOTA by 0.41%, setting a new SOTA record. Similarly, our method, despite having a 0.39% and 0.1% disadvantage in NER performance respectively on SciERC and ADE, achieved performance improvements of 10% and 0.52%. All these improvements demonstrate that two-rounds semantic interactions indeed further utilizes the predicted entities from the NER phase, significantly improving the performance of relation recognition with slightly compromising NER performance.

 doi:10.20944/preprints202505.1990.v1

Table 1. The main overall results, which highlight our new SOTA results with bold and sub-optimal performance with underline. Besides, ⋆, ○, ● respectively indicates decoder-only, encoder-decoder and encoder-only framework.

	Models	Backbone	NER			RE			RE+		
			P	R	F1	P	R	F1	P	R	F1
ACE05	SPAN (2020)	● Bert-base	89.32	89.86	89.59	-	-	-	71.22	60.19	65.24
	UniRE (2021)	● Bert-base	88.80	88.90	88.80	-	-	-	67.10	61.80	64.30
	PURE (2021)	● Bert-base	-	-	90.20	-	-	67.70	-	-	64.60
	PL-Marker (2022)	● Bert-base	-	-	89.70	-	-	68.80	-	-	66.30
	HIORE (2023)	● Bert-base	-	-	89.60	-	-	-	-	-	65.80
	HGERE (2023)	● Bert-base	-	-	89.60	-	-	-	-	-	65.80
	Mirror (2023)	○ DeBERTa-v3	-	-	86.72	-	-	-	-	-	64.88
	GPT-NER (2023)	⋆ GPT3	72.77	75.51	73.59	-	-	-	-	-	-
	ChatGPT (2023)	⋆ ChatGPT	-	-	-	-	-	-	-	-	40.50
	SET (2023)	○ T5-large	-	-	-	-	-	-	-	-	65.90
	BR (2023)	● Albert	-	-	**90.80**	-	-	-	-	-	66.00
	ATG (2024)	○ DeBERTa-v3	-	-	90.10	-	-	68.70	-	-	66.20
	BiDArtER (2024)	● Albert	-	-	89.80	-	-	-	-	-	68.40
	LAI-Net (Ours)	● Bert-base	**90.28**	**90.60**	90.44	**73.80**	**70.42**	**72.06**	**71.96**	**68.67**	**70.27**
SciERC	DyGIE++ (2019)	● SciBert	-	-	67.50	-	-	-	-	-	48.40
	Spert (2019)	● SciBert	**70.87**	69.79	**70.33**	-	-	-	53.40	48.54	50.84
	UniRE (2021)	● SciBert	65.80	**71.10**	68.40				37.30	36.60	36.90
	PURE (2021)	● SciBert	-	-	68.20	-	-	50.10	-	-	36.70
	PL-Marker (2022)	● SciBert	-	-	69.90	-	-	52.00	-	-	40.60
	HIORE (2023)	● SciBert	-	-	68.20	-	-	-	-	-	38.30
	Mirror (2023)	○ DeBERTa-v3	-	-	-	-	-	-	-	-	36.66
	ChatGPT (2023)	⋆ ChatGPT	-	-	-	-	-	-	-	-	25.90
	InstructUIE (2023)	⋆ FlanT5-11B	-	-	-	-	-	-	-	-	45.15
	SET (2023)	○ T5-large	-	-	-	-	-	-	-	-	35.90
	ATG (2024)	● SciBert	-	-	69.70	-	-	51.10	-	-	38.60
	BiDArtER (2024)	● SciBert	-	-	69.40	-	-	-	-	-	39.90
	LAI-Net (Ours)	● SciBert	70.04	69.89	69.94	**65.56**	**68.48**	**66.99**	**59.84**	**62.01**	**60.88**
ADE	Spert [**?**]	● Bert-base	89.02	88.87	88.94	-	-	-	78.09	80.43	79.24
	Table-Sequence [**?**]	● Bert-base	-	-	89.70				-	-	80.10
	SPAN [**?**]	● Bert-base	**89.88**	**91.32**	**90.59**	-	-	-	79.56	81.93	80.73
	LAI-Net (Ours)	● Bert-base	89.78	91.24	90.49	**80.48**	**83.79**	**82.09**	79.37	**83.28**	**81.25**

 doi:10.20944/preprints202505.1990.v1

4.1.2. Results Against Significant Hyperparameters

Table 2 delineates the impact of varying significant hyperparameter of G^{4}CN on the performance of LAI-Net. For the number of GCN layers, we consider a range from 0 to 5, with zero indicating the absence of GCN for semantic interaction and more layers corresponding to increased communication times among nodes within the graph. As for the number of attention heads, we opt for values of 1, 2, 3, 4, and 6. The deliberate exclusion of the value 5 is attributed to the fact that the encoder's hidden dimension is not divisible by 5, a constraint inherent to the multi-head attention mechanism.

Table 2. The ablation F1 result against: number of GCN layer, number of attention head, and whether embed entity filter or not.

Task		Number of GCN Layer						Number of Attention Head					Entity Filter	
		0	1	2	3	4	5	1	2	3	4	6	w	w/o
ACE05	NER	90.23	89.95	**90.44**	89.92	90.03	89.94	90.16	90.21	90.30	**90.44**	90.24	**90.44**	88.70
	RE	68.05	68.38	**72.06**	69.37	69.26	69.53	71.75	72.06	**72.16**	71.84	71.37	-	-
	RE+	65.61	65.75	**70.27**	66.93	66.70	66.88	69.54	**70.27**	70.03	69.86	69.21	-	-
ADE	NER	**90.49**	90.18	90.23	90.32	90.28	90.17	-	-	-	-	-	**90.49**	89.92
	RE	80.99	**82.09**	81.64	80.71	81.25	81.04	81.42	81.79	**82.09**	81.21	80.94	-	-
	RE+	80.99	**81.25**	80.81	80.39	80.95	80.63	80.83	81.02	**81.25**	80.88	80.55	-	-
SciERC	NER	69.40	**69.94**	69.47	69.17	69.31	69.40	69.42	69.76	69.32	**69.94**	69.23	**69.94**	69.76
	RE	66.08	66.27	**66.99**	66.43	66.35	66.28	65.80	65.66	65.62	**66.99**	64.48	-	-
	RE+	60.49	60.57	**60.88**	59.91	60.82	60.06	60.24	60.15	60.61	**60.88**	59.56	-	-

The table permits an intuitive observation that: (1) an increase in the GCN layers number does not linearly translate to enhanced performance; an excessive GCN layers number can exert a deleterious effect on the model, with the more optimal layers number identified as either 1 or 2; (2) concerning the number of attention heads, the more optimal solution exhibits some variation across different datasets, yet it is unequivocally clear that neither an excessively high (e.g. 6) nor a disproportionately low (e.g. 1) number of attention heads can fully capitalize on the GCN's capabilities.

4.2. Inference Speed

There is justification to assess whether the adoption of GCNs to enhance F1 performance has made trade-offs respect to inference efficiency, for the reason that GCNs have certain efficiency disadvantages typically. Hence, We conducted a comparison between PL-Marker and LAI-Net in terms of inference speed, that experiment was evaluated on a GeForce RTX 3090 24GB GPU with a evaluate batch size of 16. We use Bert-base model for ACE05 and SciBert for SciERC.

As shwon in Table 3, we indeed sacrificed varying degrees of inference efficiency in exchange for varying degrees of improvement in F1 performance. Overall, the greater the sacrifice, the greater the benefit.

Table 3. The inference speed comparison results.

Task		Metric	PL-Marker	LAI-Net
ACE05	NER	F1	89.70	**90.44**
		Speed (sent/s)	**62.94**	35.10 (-44.23%)
	RE	F1	66.30	**70.27**
		Speed (sent/s)	**93.14**	43.00 (-53.83%)
SciERC	NER	F1	69.90	**69.94**
		Speed (sent/s)	**54.13**	52.17 (-3.62%)
	RE	F1	40.60	**60.88**
		Speed (sent/s)	**93.29**	39.57 (-57.58%)

 doi:10.20944/preprints202505.1990.v1

4.3. Ablation Study

4.3.1. Ablations Against Entity Filter

Considering the binomial surge in candidate entity quantity accompanying increased entity length in NER phase, the relatively paltry positive examples can be easily overwhelmed by a vast array of negative samples, thereby instilling the network a strong propensity to categorize all samples as negative, leading to difficulty in accurately identifying genuine entities. To mitigate this, LAI-Net incorporates a deliberately inserted filter prior to the entity classifier to preliminarily sieve out spans that are clearly non-entity. To validate whether said filter genuinely facilitates the NER process, we devised associated ablation experiments. As depicted in the two rightmost columns of Table 2, the presence of the filter effectively improves NER performance, with advantages most conspicuous on the ACE05 dataset (surpassing no-filter models by 1.74% in terms of F1 scores) and improvements of varying degrees are also observed on other datasets.

4.3.2. Ablation Against Two Rounds of Interaction

We conducted ablation experiment specifically targeting the two-round semantic interaction. Drawing upon the outcomes of the experiment, we can directly evaluate the extent to which our devised dual semantic interaction genuinely augments the model's performance. What should be noted is that the second round interaction is built upon the annotated label information (that's what the first round interaction do), so the second round interaction ceases to exist once the label annotation information is no longer injected. However, the existence of the second round interaction does not affect the first round interaction. As shown in Table 4, no matter which round of semantic interaction is eliminated, it invariably leads to adverse effects of varying magnitudes on the newly SOTA we have developed, encompassing the precision, recall, and F1-score metrics. From the perspective of performance degradation, the detrimental effect of eliminating the second round interaction on the basis of LAI-Net is significantly more pronounced than the impact of further removing the first round of interaction on the basis of w/o 2nd round interaction, with the discrepancy peaking at over 21-fold ($\frac{4.67\%}{4.89\%-4.67\%}$ for RE+ in ACE05).

 doi:10.20944/preprints202505.1990.v1

Table 4. The ablation against two rounds interaction.

Task		Method	P	R	F1
ACE05	NER	LAI-Net	90.28	90.60	**90.44**
		w/o 2nd	89.97 (-0.31)	90.50 (-0.10)	90.23 (-0.21)
		w/o 1st	89.72 (-0.55)	90.68 (0.08)	90.20 (-0.24)
	RE	LAI-Net	73.80	70.42	**72.06**
		w/o 2nd	69.70 (-4.09)	66.49 (-3.94)	68.05 (-4.01)
		w/o 1st	68.64 (-5.16)	67.16 (-3.26)	67.89 (-4.17)
	RE+	LAI-Net	71.96	68.67	**70.27**
		w/o 2nd	67.20 (-4.77)	64.10 (-4.58)	65.61 (-4.67)
		w/o 1st	66.47 (-5.49)	64.35 (-4.32)	65.39 (-4.89)
ADE	NER	LAI-Net	89.78	91.24	**90.49**
		w/o 2nd	-	-	-
		w/o 1st	88.94 (-0.84)	91.07 (-0.17)	89.99 (-0.51)
	RE	LAI-Net	79.38	83.29	**81.26**
		w/o 2nd	79.01 (-0.37)	83.17 (-0.12)	81.04 (-0.22)
		w/o 1st	79.08 (-0.30)	82.99 (-0.30)	80.99 (-0.27)
	RE+	LAI-Net	79.37	83.28	**81.25**
		w/o 2nd	78.73 (-0.64)	82.64 (-0.64)	80.63 (-0.62)
		w/o 1st	78.47 (-0.91)	82.44 (-0.84)	80.41 (-0.85)
SciERC	NER	LAI-Net	70.04	69.89	**69.94**
		w/o 2nd	69.82 (-0.22)	68.98 (-0.91)	69.40 (-0.54)
		w/o 1st	69.58 (-0.46)	69.12 (-0.77)	69.35 (-0.59)
	RE	LAI-Net	65.56	68.48	**66.99**
		w/o 2nd	64.21 (-1.35)	68.07 (-0.41)	66.08 (-0.91)
		w/o 1st	63.96 (-1.60)	67.82 (-0.66)	65.83 (-1.16)
	RE+	LAI-Net	59.84	62.01	**60.88**
		w/o 2nd	59.79 (-0.05)	61.22 (-0.80)	60.49 (-0.39)
		w/o 1st	59.81 (-0.03)	60.47 (-1.54)	60.14 (-0.74)

4.3.3. Ablations Against Attention Mask Matrix

In order to evaluate the efficacy of partial attention matrices, we selected three distinct attention mask matrices for ablation (detail explaned in Appendix B.11), namely: the full attention matrix, wherein all tokens are mutually visible; the anno-token visible attention matrix, where annotation tokens and text tokens are intervisible; and the anno-token invisible attention matrix, with annotation tokens and text tokens also being intervisible. Irrespective of the attention masking matrix employed, tokens of the same type remain mutually visible during the computation of attention scores.

As Table 5 elucidates, across all three datasets, the anno-token invisible attention matrix exhibits a markedly superior performance with strict F1 metrics compared to the other attention matrice types. The anno-token visible attention matrix comes in second place, with its largest deficit compared to the top-performing technique capping out at 4.7% across the varied tasks encapsulated within the trio of benchmarks. Meanwhile, the fully attention matrix acts a performance lagging behind the peak achieved score on each respective dataset (max up to 6.28%), indicative of appreciably inferior capabilities amongst the range of workloads tested.

We attempt to analyze the underlying reasons, which may lie in the following points: (1) the mutual visibility mechanism between annotation tokens and text tokens establishes a conduit for semantic communication, thereby enhancing the semantic richness of the embeddings for both annotations and text. On the one hand, this enables annotation tokens to fully comprehend background information of text, which directly impacts their high-dimensional semantic representation. On the other hand, text tokens can also fully integrate the information from annotations, which directly facilitates the screening and categorization of candidate entities. (2) conversely, the full attention matrix allows for the intermingling of semantic information among annotation tokens, text tokens, and entity candidate tokens, which may lead to an excessive infusion of additional semantics, resulting in semantic redundancy, and ultimately causing a decline in model performance.

 doi:10.20944/preprints202505.1990.v1

Table 5. The ablation result against attention mask matrix, where Vis. represents visible, Inv. represents invisible.

Task		NER	RE	RE+
ACE05	Inv.	**90.44**	**72.06**	**70.27**
	Vis.	90.36 (-0.08)	68.01 (-4.05)	65.57 (-4.70)
	Full	88.29 (-2.14)	65.79 (-6.28)	64.28 (-5.99)
ADE	Inv.	**90.49**	**82.09**	**81.25**
	Vis.	90.09 (-0.40)	81.26 (-0.83)	80.09 (-1.16)
	Full	89.40 (-1.10)	79.99 (-2.10)	78.92 (-2.33)
SciERC	Inv.	**69.94**	**66.99**	**60.88**
	Vis.	69.13 (-0.81)	66.44 (-0.55)	60.61 (-0.27)
	Full	66.21 (-3.73)	66.06 (-0.93)	60.30 (-0.59)

5. Conclusion

Faced with the challenges posed by sub-task excessive separation and modeling gaps between entities and relations, we propose LAI-Net, which leverages a two-phase semantic interaction and attains SOTA performance in both NER and RE tasks. The key novelty is the multi-phase semantic interaction framework to effectively inject external knowledge and unify representations for entities and relations. We manually annotate label abbreviations to fully-semantic unbroken phrase for expand lexical semantic, and then leverages G^4CN to fuse information from latent multi-view. And an entity candidate filter is embed to coarse screening of candidate spans for NER. Experiments on several datasets show that our LAI-Net allows high-level semantic information flow and facilitates the E2ERE task, successfully achieves the SOTA performance.

Author Contributions: Rongxuan Lai participated in and led all aspects of the research, including but not limited to data analysis and preprocessing, algorithm design, code implementation, experimental procedure formulation, analysis of experimental results, writing and reviewing of the paper. Zhenyi Wang and Haibo Mi contributed to the algorithm design, conducted the data analysis, and critically reviewed the manuscript for important intellectual content. Wenhui Wu and Li Zou was responsible for the statistical analysis and interpretation of data, and also helped in revising the manuscript for better clarity and coherence. Feifan Liao participated in the data collection and contributed to the writing of the sections related to methodology and results. All authors have read and approved the final manuscript and have agreed to be accountable for all aspects of the work in ensuring that questions related to the accuracy or integrity of any part of the work are appropriately investigated and resolved.

Data Availability Statement: The datasets used during the current study in this article are available publicly in Appendix B.1. Access to these data is open to all interested researchers, and any restrictions on data availability, such as those related to participant privacy or confidentiality can be found in the websites mentioned in Appendix B.1. The processing of the data and the associated code will be completely transparent after the paper is accepted.

Acknowledgments: This work was supported by National Key R&D Program of China (Grant No.2024ZD01NL00100). The authors declare no other relevant financial or non-financial interests related to the content of this manuscript that could be perceived as influencing the work reported.

Conflicts of Interest: This research was funded by the project with the grant numbers 2024ZD01NL00100. The authors declare no other relevant financial or non-financial interests related to the content of this manuscript that could be perceived as influencing the work reported. This statement is provided to ensure a transparent and unbiased evaluation of the research.

Appendix A Vanilla Graph Convolutional Network

Typically, a good text representation is a prerequisite for achieving superior model efficacy, which has motivated numerous researchers to try to leverage contextual features through diverse model architectures to better comprehend textual semantics, thereby enhancing the representation of text. Among all these architectural choices, Graph Convolutional Network (GCN), whose goal is to learn structure-aware node representations on the graph, is a widely utilized framework for encoding

 doi:10.20944/preprints202505.1990.v1

information within graphs, where in each GCN layer, every node engages in information exchange and communication with neighboring nodes through their connections (called edges). The efficacy of the GCN model in encoding contextual information on a graph of input sentences has been demonstrated by numerous prior studies.

Now, we briefly describe the vanilla graph convolutional networks in mathematical form. The first step in utilizing GCNs is to build a graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ from plain text, where $v_i \in \mathbf{V}$ represents a node of $\mathcal{G}$, and $e_{ij} = (v_i, v_j)$ represents the edge between nodes v_i and v_j, respectively. The structural information that GCN can comprehend is concretely manifested in sentences as the dependencies between words, which is represented by an adjacency matrix $\mathbf{A} = (a_{i,j})_{n \times n}$. In vanilla GCN, $a_{i,j} = 1$ if $i = j$ or there is a dependency connection (arc) between two words/tokens x_i and x_j, and $a_{i,j} = 0$ otherwise. However, in our work, we adopt a newly method to initiates the $a_{i,j}$ with neural networks.

Based on the graph with adjacency matrix $\mathbf{A}$, for each node v_i, the $l-$th GCN layer gathers the information carried by its neighbor node and computes the output representation $\mathbf{h}_i^{(l)}$ for v_i by:

$$\mathbf{h}_i^{(l)} = \sigma\left(\sum_{j=1}^{n} a_{i,j}\left(\mathbf{W}^{(l)} \cdot \mathbf{h}_j^{(l-1)} + \mathbf{b}^{(l)}\right)\right) \tag{A1}$$

where σ is the activation function, and $\mathbf{h}_j^{(l-1)}$ denotes the output representation of v_j from the $(l-1)$-th GCN layer, $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are trainable matrices and the bias for the l-th GCN layer, respectively.

In the other words, the l-th layer in a vanilla GCN can be written in the form of matrices as follows:

$$\hat{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}} \tag{A2}$$

$$\mathbf{H}^{(l+1)} = \sigma(\hat{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}) \tag{A3}$$

$$\mathbf{H}^{(l)} = \{\mathbf{h}_1^{(l)}, \mathbf{h}_2^{(l)}, \mathbf{h}_3^{(l)}, \cdots, \mathbf{h}_{|\mathbf{V}|}^{(l)}\} \tag{A4}$$

with $\mathbf{H}^{(0)}$ is the combination of all initial node representations, where $\hat{\mathbf{A}}$ is the normalized symmetric adjacency matrix and $\mathbf{W}^{(l)}$ is a parameter matrix for the l-th GCN layer, $\mathbf{D}$ is the diagonal node degree matrix, where $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$, σ is a non-linear activation function like ReLU.

Finally, the entire L-layer vanilla GCN modeling process can be formulated as:

$$\begin{aligned}\mathbf{GCN}\left(\mathbf{A}, \mathbf{H}^{(0)}\right) &= \mathbf{H}^{(L)} \\ &= \sigma(\hat{\mathbf{A}}\mathbf{H}^{(L-1)}\mathbf{W}^{(L-1)})\end{aligned} \tag{A5}$$

Figure A1. Statistical analysis diagram of algorithm stability, in which each data point represents the averaged F1 performance with various random seeds, and the shaded area surrounding the data line indicates the error range under the specified hyperparameters (GCN layer number or attention head number), which we measure with standard deviation.

Appendix B Implement Details

Appendix B.1 Datasets and Preprocess

We selected two standard corpus (ACE05, SciERC) in terms of E2ERE task.

1) ACE05[2] is collected from a variety of domains (such as newswire and online forums). It includes 7 entity types, and include 6 relation types between entities. For data processing, we use the

[2] https://catalog.ldc.upenn.edu/LDC2006T06

 doi:10.20944/preprints202505.1990.v1

same entity and relation types, data splits[3], and pre-processing as [**?** **?**] (351 training, 80 development and 80 testing).

2) SciERC [**?**] is a scientific-oriented dataset, which is built from 12 AI conference/workshop proceedings in four AI communities, and includs 7 entity types and 7 relation types.

In terms of experiment, we run our model 10 times with different random seeds, and report averaged results of all the runs.

Appendix B.2 Chosen of Baselines

To ensure fairness and comprehensiveness in performance comparison as much as possible, we have established several principles for selecting baselines The first is to select well-recognized baselines from those cited in the SOTA works published in recent years. The second principle is to cover models of encoder-only, decoder-only and encoder-decoder architectures as comprehensively as possible, and what the priority is given to models that have similar or comparable parameter scales to our PLM during the selection process for encoder-only based baselines. Thirdly, one or more datasets should be the same as those used in our work. Lastly, we tend to select methods that formally published in relatively authoritative conferences or journals, which is not mandatory.

Appendix B.3 PLMs and Hardware Devices

For fair comparison with previous works, we employ bert-base-uncased [**?**] as the encoders for ACE05, and use the in-domain scibert-scivocab-uncased[4] [**?**] as the encoder for SciERC, and all the experiments are executed using three GeForce RTX 3090 24GB GPUs.

Appendix B.4 Optimizer and Learning Rate Settings

We use AdamW optimizer during training. We set the learning rate as 4e-4 for both NER and RE task. We had tried to set different learning rate for different layers, and experiment results show that it's useless.

Appendix B.5 Maximum Length Settings

We respectively set the maximum length of reorganized sentence C as 150, 150 on ACE05, SciERC. As enumerating possible spans, we set the maximum span length L as 8 for all datasets, and limit the number of entity candidates as 220 for every train/eval sample.

Appendix B.6 Batch Size and Epoch Settings

In NER phase, we respectively set batch size as 16 per GPU for SciERC, 20 per GPU for ACE05. And in RE phase, we set the batch size as 40 per GPU for SciERC, 50 per GPU for ACE05. We set the epoch as 80 for all Datasets in NER phase, and 60 for all dataset in RE phase.

Appendix B.7 Avoidance the Negative Influence of Annotation

Considering that our method increases the length of the training instances, which could impact the model's performance, we strive to alleviate this negative impact as much as possible. Firstly, avoid using overly long annotation information, but simply restore label abbreviations(such as restoring the PER tag to "person" and the PER-ORG tag to "person-organization"), and the length of the annotations used in the experiments does not exceed 10 words. Secondly, using adjusted attention mask matrices to minimize semantic confusion caused by the insertion of annotations.

[3] https://github.com/tticoin/LSTM-ER/tree/master/data/ace2005/split

[4] SciBERT is a BERT model trained on scientific text, whose corpus includes the full text of 1.14 million scientific papers (82% in biomedical and 12% in computer science), and may be more suitable for natural language processing tasks on SciERC dataset

 doi:10.20944/preprints202505.1990.v1

Figure A2. Diagrams of three types attention mask matrix with an assumption that there is two entity candidates.

Appendix B.8 Cold Start Settings for NER

It should be noted that, due to the enumeration of a large number of non-entity spans as negative samples in NER pahse, which extremely likely to lead to unable-convergence train, or the phenomenon that the model directly predicts all spans as non-entities. Therefore, we set up a specific cold start process for training. In the first half of training, we use the true labels as the filter result for next entity classification in order to calculate the loss to update the parameters. This guides the model during early learning before exposing it to negatively labeled non-entity spans. In actual training, we set the cold boot epoch number as 15, 40 for ACE05, SciERC respectively. Moreover, according to unpublished experimental results, experiments without cold start configuration were utterly incapable of acquiring any knowledge whatsoever, with all performance metrics equaling zero throughout the training phase.

Appendix B.9 Symmetry of Relation for RE

We formulate the directed relation as $r_{i,j}$, with the subject entity e_i always pointing to object entity e_j. Therefore, a triplet with positive relation canbe written as $e_i \rightarrow r_{i,j} \rightarrow e_j$, and its reverse formula is $e_j \rightarrow r_{j,i} \rightarrow e_i$. $r_{i,j}, r_{j,i}$ may refer to different relation types. In RE phase, we consider the subject be left and the object be right by default. Either $(e_i \rightarrow r_{i,j} \rightarrow e_j)$ or $(e_j \rightarrow r_{j,i} \rightarrow e_i)$ will be predicted if it's a really relaiton. Only when they are both predicted to be true (that is, the probability value is greater than the threshold), the triplet $(e_i, r_{i,j}, e_j)$ and triplet $(e_j, r_{j,i}, e_i)$ will be established.

Appendix B.10 Stability of Training

To ensure the reliability of the experimental data, we repeated the experiments multiple times for each hyperparameter combination, each time initializing the network with a different random seed, and then calculated the mean and standard deviation of the experimental results, as shown in the Figure A1. The first row of chart in the Figure A1 represents the fluctuations in NER and RE performance across three datasets for different numbers of GCN layers, The second row of chart in the Figure A1 represents the fluctuations in NER and RE performance across three datasets for different numbers of attention heads.

Appendix B.11 Parital Attention Mask

Obviously, the series of annotation tokens and marker tokens manually attached after the text tokens does not form a coherent and semantically complete sentence. It inevitably affects the semantic construction from text tokens, and impair the representational ability of word vectors during the PLM encoding process.

To address this potential issue, we adopt a specialized partial attention mechanism to selectively mitigate or enhance the semantic impact of tokens from differed types. In details, by adjusting the value of elements of the attention mechanism mask matrix, we can control the visibility among three types of tokens.

Partial attention can effectively control the information flow between different tokens. It suppresses the information interaction between text and annotations (i.e. invisible) while enhancing the information interaction between text and candidates (i.e. visible).

We show three different masking matrices in Figure A2, for which we conduct some ablation experiments. In conjunction with Figure A2, we further elucidate the meaning of the discrete elements within mask matrix. Train our gaze upon the first row of Figure A2 (a), which delineates the tokens discernible by the text token. The pale green region signifies it can see corresponding tokens, the white space those it cannot see, and the faint yellow elements the marker tokens within its purview. Notably, the two faint yellow squares on the left denote the start marker tokens, while the two on the right denote the end marker tokens.

 doi:10.20944/preprints202505.1990.v1