IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 24 August 2024, accepted 27 October 2024, date of publication 30 October 2024, date of current version 18 November 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3487987

RESEARCH ARTICLE

Agent-Like Model for the Fusion of Attention Features for the Extraction of Joint-Entity Relations

JIM-WEI WU[1], (Member, IEEE), HANG-KAI YE[1], JIA-CHENG LI[1], AND JUNG-YU LIAO[2]
[1]Department of Electrical Engineering, National Central University, Taoyuan City 320317, Taiwan
[2]Department of Health Promotion and Health Education, National Taiwan Normal University, Taipei City 106209, Taiwan

Corresponding author: Jung-Yu Liao (jyliao@ntnu.edu.tw)

This work was supported by the National Science and Technology Council of Taiwan under Grant NSTC-113-2221-E-008-084 and Grant NSTC-113-2410-H-003-043.

ABSTRACT Relation extraction involves identifying related entity pairs within sentences and matching them with corresponding relation types. This paper introduces an agent-like model that fuses attention features to facilitate relation extraction. Based on a cascade binary tagging framework, the model uses an agent-like module enabling the efficient extraction of relations and implicit semantic information from training data. In experiments, the proposed model improved efficiency in extracting relational triples.

INDEX TERMS Relation extraction, agent-like model, deep learning, extraction efficiency.

I. INTRODUCTION

In knowledge graph construction and other applications, the process of extracting usable information from sentences depends largely on relation extraction, which involves identifying related entity pairs within sentences for matching with the corresponding relation types. Clearly, relation extraction involves entity recognition as well as classification.

The information required for relation extraction is known as a relational triple, comprising a *subject, a relation, and an object* [1]. The term *subject* refers to the first entity obtained from the text, while the term *object* refers to an entity sharing at least one specific relationships with the subject. The term *relation* describes the nature of the relationship that exists between the subject and object.

In early research on relation extraction, researchers generally adopted a 2-module pipeline approach [2], involving relational entity identification followed by relational entity classification. This approach is simple, direct, easy to implement, and easy to debug. Nonetheless, despite its modularity and flexibility, the pipeline method is prone to error propagation, and the parameters of the two modules cannot be shared with each other, which limits model training performance.

These limitations prompted the development of joint modeling and parameter-sharing approaches to relation extraction. This involved redesigning the internal architecture of the relation extraction model to combine the entity extraction and relationship classification subtasks, allowing end-to-end training. This enhanced both entity recognition and relationship classification capabilities, leveraging the correlation between them to improve overall performance.

Extensive research in joint entity and relation extraction has resulted in the development of numerous extraction frameworks [3], [4], [5], [6], [7], [8]. Although the use of two prediction subtasks adds complexity, this approach provides considerable flexibility in the design of the internal framework [9], [10], [11], [12], [13], [14], [15].

This paper introduces an agent-like model that fuses attention features to improve prediction performance for relation extraction. The encoder module generates an agent parameter containing semantic information related to relationship categories and sentence context. This parameter is then used to calculate attention weights with separate vectors for relationship category and sentence. A linear attention mechanism produces an encoding vector that fuses semantic information.

The associate editor coordinating the review of this manuscript and approving it for publication was Lorenzo Mucchi.

Finally, subject and object taggers are applied to obtain the relation triples present in the sentence.

Section II details the relation extraction task and outlines the basic principles on which the proposed method is based. Section III details the extraction method for obtaining triples from sentences and corresponding objective function. Section IV explains the design of the proposed model and loss function used for model training. Section V introduces the public datasets used in our experiments on relation extraction, followed by our results and model performance comparisons. Conclusions are drawn in Section VI.

II. RELATED WORKS

A. OVERLAPPING TRIPLE CLASSIFICATION

The overlapping triple problem is a special scenario in relation extraction that can be observed from three perspectives: 1) Multiple relational triples sharing one or more components within the same sentence; 2) Multiple triples involving the same entities but forming entity pairs with different relationship types; and 3) One entity forming pairs with multiple different partners across different triples.

Consider the following example: ''Test pilot Alan Bean, who is now retired, was born in Wheeler Texas''. In this sentence: ''Alan Bean'' and ''pilot'' form an entity pair with the relationship ''occupation''. At the same time, ''Alan Bean'' and ''Texas'' form an entity pair with the relationship ''birthplace''. The accurate extraction of all relation triples in the text requires that the Relation Extraction (RE) model repeatedly consider these special entities during processing. This involves detecting all related tail entities and identifying the corresponding relationship types. This presents a serious challenge.

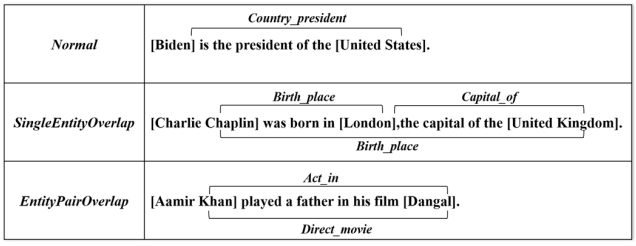

FIGURE 1. Illustrative instances of relational triples: ***Normal*** **(non-overlapping),** ***Single-entity overlap*** **(SEO), and** ***Entity-pair overlap*** **(EPO).**

As shown in Fig. 1, relational triples can be classified according to the number of overlapping entities. Scenarios involving no overlapping entities are referred to as *normal*. Scenarios involving a single overlapping entity are referred to as *Single-entity Overlap* (SEO), while those with overlapping entity pairs are referred to *Entity-pair Overlap* (EPO). In EPO scenarios, multiple relationships types could exist between the same entity pairs.

B. AGENT ATTENTION MECHANISM

When used as the activation function in the attention mechanism, the softmax function [16] converts attention scores from the query and key into probability distributions, while ensuring that all attention weights fall within the [0,1] interval and sum to 1. The properties of the softmax function ensure that the model outputs clear semantic explanations to facilitate the comprehension and interpretation of the prediction results. Nonetheless, attention mechanisms that rely on the softmax function are prone to high computational complexity. The agent attention method proposed by Han et al. [17] is meant to strike a reasonable balance between computational efficiency and representation capability.

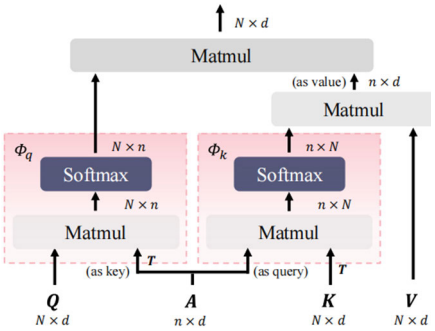

FIGURE 2. Architecture of agent attention [17].

The agent attention method introduces a fourth parameter, Agent, along with the *Query*, *Key*, and *Value* used in the original attention mechanism. The structure of this 4-parameter attention paradigm (Q, A, K, V) is illustrated in Fig. 2. Unlike conventional attention mechanisms, attention parameters Q and K are not used directly to calculate the attention score. Instead, an agent vector (A) is generated by incorporating information from both Q and K to facilitate the transfer of information between them.

Two softmax attention modules are constructed using (*Query*, *Agent*) and (*Agent*, *Key*) to calculate the attention weight of Q to A and A to K, respectively. Once the two attention calculations are completed, the two softmax attention modules are used as two components (Linear attention and *Value*) to form a linear attention module. The attention score between A and K is calculated by the softmax function, yielding the corresponding attention weight. This allows us to adjust Value by calculating the product of Value and the transpose of A and K. Finally, the adjusted Value is combined with Q to generate the final output.

Essentially, by combining two conventional softmax attention operations in a manner akin to generalized linear attention, agent attention enhances the representational capacity of the model, while reducing computational complexity.

III. RELATION ENTITY EXTRACTION

This chapter outlines each element of the relational triple and defines a training objective function for the joint entity-relation extraction task, in accordance with the framework proposed by Wei et al. [3].

Our core objective in joint-entity relation extraction is to extract the relational triple (subject, relation, object) via

subject tagging and relation-specific object tagging, in accordance with the extracted subject.

A. SUBJECT TAGGING

The first step involves a sequence of labeling tasks aimed at identifying the sentence subject for use in extracting relational entity pairs. During the subject tagging operation, each word in the sentence is processed sequentially by the model, which assigns a probability indicating the likelihood that the word represents the start or end of the subject.

After obtaining the probability of each token in each sequence, a hyperparameter is set to define a probability threshold. If the probability of a token exceeds this threshold, then the model tags it with a value of 1, marking it as the start or end of the subject. If the predicted probability falls below the threshold, then the current token is tagged with a value of 0.

As shown in Fig. 3, the model then extracts the spans between each start and end token in the output sequence in order to identify all of the subjects in the sentence.

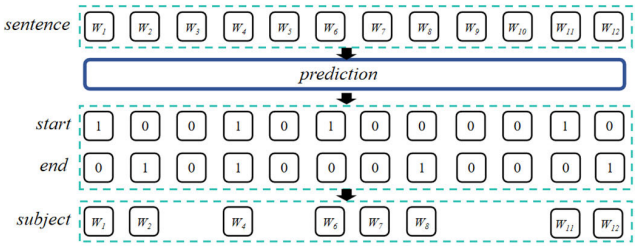

FIGURE 3. Process of subject extraction.

After extracting the subject, we derive the objective optimization function for subject extraction based on the prediction results related to the start and end sequences, as follows:

$$P_{\theta_s}(s \mid x) = \prod_{t \in \{start\_s, end\_s\}} \prod_{i=1}^{L} (p_{sub}^{i})^{\mathrm{I}\{y_i^t=1\}} (1 - p_{sub}^{i})^{\mathrm{I}\{y_i^t=0\}} \quad (1)$$

where θ_s indicates parameters from the subject extraction process; I{· } represents the true tagging of the subject in sentence x; y_i^t indicates the tag in the i-th token; L indicates the length of sentence x; P_{sub}^{i} indicates the probability of accurately predicting the start and end positions of the object.

B. RELATION-SPECIFIC OBJECT TAGGER

The next step involves identifying the object with a specific relationship to each extracted subject. The process of object extraction is similar to that of subject extraction, except that in the input sequence, the model first fuses the sentence sequence with the corresponding subject representation vector in the input sequence.

As shown in Fig. 4, the model iterates through all predefined relation categories to enable the extraction of the objects

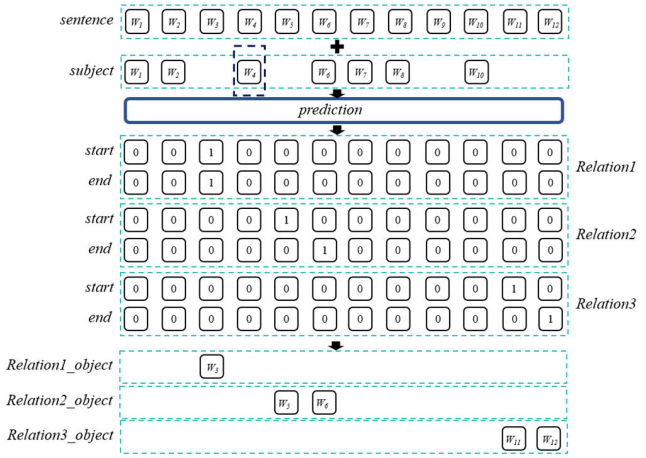

FIGURE 4. Process of relation-specific object extraction.

that correspond to each category of interest. This makes it possible to extract all relation triples present in the sentence.

After extracting all of the objects, we derive the objective optimization function based on the prediction results pertaining to the start and end sequences under various subject and relation categories, as follows:

$$P_{\theta_o}(o \mid x, s, r) = \prod_{t \in \{obj\_s, obj\_e\}} \prod_{i=1}^{L} \left(p_{obj}^{i}\right)^{\mathrm{I}\{y_i^t=1\}} \left(1 - p_{obj}^{i}\right)^{\mathrm{I}\{y_i^t=0\}} \quad (2)$$

where θ_o refers to parameters used in object extraction; I{· } indicates the true tagging of the object from sentence x; L indicates the length of sentence x; P_{obj}^{i} indicates the probability of accurately predicting the start and end positions of the object. Note that if no object is available for the subject, then we assign the tag $y_i^t = 0$ in I{·}.

C. OBJECTIVE OPTIMIZER FUNCTION

After extracting the relation triples, the model undergoes iterative training to improve its accuracy in this task. This allows the use of objective optimization functions for both subject and object extraction to derive an objective optimization function for the overall model.

The main goal of the relation extraction task is to extract complete relational triples. Thus, the objective function of the model should be set to maximize the probability of correctly extracting all relation triples in the dataset. Later, the relational triple of the dataset can be decomposed into extractions for every subject in the sentence and their relation-specific corresponding objects, which are then used to derive the overall objective optimization function.

$$\prod_{(s,r,o) \in T} P((s, r, o) \mid x) = \prod_{s \in T} P_{\theta_s}(s|x) \prod_{(r,o) \in T} P((r, o)|x, s)$$

$$= \prod_{s \in T} P_{\theta_s}(s \mid x) \prod_{r \in T|s} P_{\theta_o}(o \mid x, s, r) \times \prod_{r \in R \setminus T|s} P_{\theta_o}\left(o_\phi \mid x, s, r\right) \quad (3)$$

where $s \in T$ refers to the subject from dataset T;$r \in T|s$ refers to the relation corresponding to subject s from dataset T; $r \in R\setminus T|s$ indicates all relations except the one corresponding to the subject from dataset T; and o_ϕ refers to a subject without a related object in the sentence.

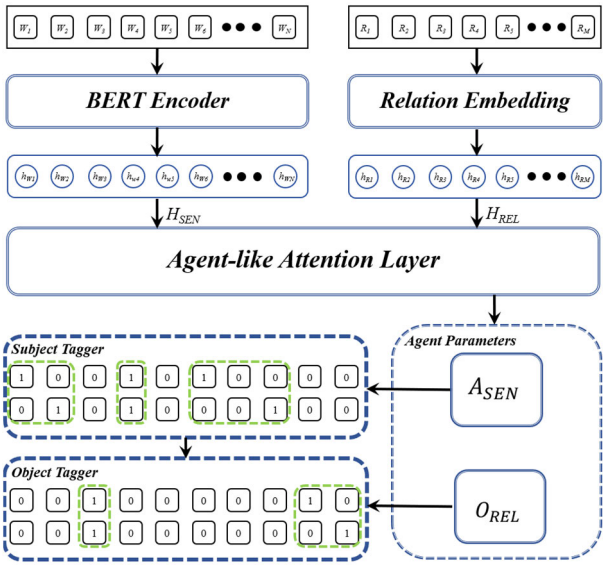

FIGURE 5. Architecture of agent-like model for the fusing of attention features.

IV. AGENT-LIKE MODEL FOR THE FUSING OF ATTENTION FEATURES

Fig. 5 illustrates the architecture of the agent-like model used in the fusing of attention features to facilitate joint-entity relation extraction. The framework comprises three main parts: A vector encoding module, an agent-like attention layer, and a sequence entity tagger.

The initial vectors are encoded by the vector encoding module, using the input sentences and predefined relationship categories. These encoded vectors, representing the sentence context and relation categories, are then processed through the agent-like attention layer, producing a final representation vector. This output is subsequently used by the sequence entity tagger to extract the subject and its corresponding relation-specific object.

A. VECTOR ENCODING MODULE

Joint-entity relation extraction is performed by inputting the sentences and predefined relationship categories used for relational triple extraction from the training dataset into the vector encoding module to obtain the initial context vector and initial relation category vector.

The BERT pre-trained language model is used to obtain the initial context vector of sentence x in the training dataset. This involves using the BERT tokenizer to segment sentence x into a sequence composed of n sub-words to obtain sub-word sequence S. In accordance with the predefined dictionary in the BERT [18] model, each sub-word in sequence S is converted into its corresponding ID number, forming ID number sequence I. Each ID number in the dictionary represents a unique sub-word corresponding to a fixed vector representing the semantics of the sub-word.

ID sequence I is fed into the pre-trained BERT model to generate vector representation H_{SEN} for the sub-word sequence. Each sub-word vector representation contains rich general semantic information and contextual content of the sentence, which can be applied to downstream relation extraction tasks using the BERT model. Vector representation H_{SEN} is used as the initial context vector for the model, as follows:

$$H_{SEN} = BERT\,(I) = [h_1, h_2, h_3, h_4, h_5, h_6, \ldots, h_i] \quad (4)$$

where $H_{SEN} \in \mathbb{R}^{(m \times d)}$ refers to the initial sentence vector from ID sequence I; m indicates the length of the initial context vector H_{SEN}; d indicates the dimensions of the vector H_{SEN}; $BERT(\cdot)$ refers to the process of feeding the input sequence into a pretrained language BERT model; and h_i is the vector representation from i-th sub-word.

Relationship categories between entity pairs are obtained from the relation extraction dataset and represented as labels. These categorical labels are embedded as high-dimensional vectors, corresponding to each predefined relationship category. These unique high-dimensional vectors are referred to as relation embeddings.

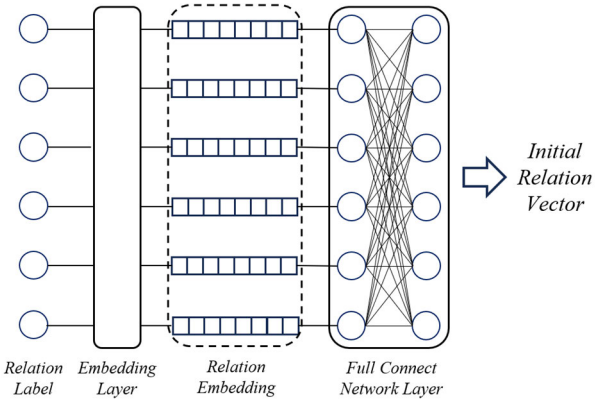

FIGURE 6. Relation category processing workflow of initial relation vector H_{REL}.

The relation embeddings are linearly transformed using the weight matrix and bias vector of a fully connected network layer. As training progresses, connection weights and bias parameters are iteratively optimized to obtain relation vectors specific to the relation extraction task. Fig. 6 outlines the relation category processing workflow. These relation embeddings are defined as the initial relation vector H_{REL} in this model.

$$K = embed\,([R_1, R_2, R_3, R_4, R_5, \ldots R_m]) = [h_1, h_2, h_3, h_4, h_5, h_6, \ldots h_m] \quad (5)$$

$$H_{REL} = W_r \cdot K + b_r \tag{6}$$

where K indicates the relationship embedding corresponding to the relationship category; $embed(\cdot)$ refer to the process of converting relationship category labels into high- dimensional vectors; h_m indicates the representation vector from the m-th relationship category; $H_{REL} \in \mathbb{R}^{(c\times d)}$ indicates the initial relation vector; d indicates the dimensions of the vector H_{REL}; c indicates the number of the predefined relationship categories; and W_r and b_r are trainable parameters.

B. AGENT-LIKE ATTENTION LAYER

In the agent-like attention layer, the initial context vector and initial relation vector (from the vector encoding module) are used as inputs. Note that the focus of this paper is the agent-like attention layer, which is responsible for learning important relational semantic information. It is based on the architecture of the agent attention model proposed by Han et al. [17].

Note that conventional attention mechanisms include three main elements [16]: Query (Q), key (K), and Value (V). The attention-weighted calculation of input information depends on learning the semantic relationships among these three parameters. The proposed agent-like attention scheme uses the same parameters as well as an additional parameter , A (the agent parameter), which incorporates both query and key semantic information.

The attention method is used to calculate agent parameter A. In this step, initial relation vector H_{REL} is used for parameter Q and the initial sentence vector H_{SEN} is used for the parameters K and V. To enhance the semantic information representation of the attention parameters, we feed the initial vectors of Q and K (or V),H_{REL} and H_{SEN}, into a linear layer to obtain the corresponding weight parameters. The Q, K, and V vectors are obtained after the linear transform, as follows:

$$\begin{aligned} Q &= W_Q \cdot H_{REL} \\ K &= W_K \cdot H_{SEN} \\ V &= W_V \cdot H_{SEN} \end{aligned} \tag{7}$$

where Q, K, and V are attention parameters; and W_Q, W_K, W_V refer to the corresponding weight matrices.

Next, we concatenate attention parameters Q and K and input them into the linear layer to obtain attention score a^{ij}_{score}, which is fed into the softmax function to obtain the corresponding attention weight a^{ij}_{score}. Finally, we calculate the sum of the weighted value, and apply a residual connection [19] operation to avoid degradation in deep networks. This process yields A_{SEN} as agent parameter A containing semantic information from both H_{SEN} and H_{REL}.

$$a^{ij}_{score} = W_a \left[W^i_Q \cdot H^i_{REL}; W^j_K \cdot H^j_{SEN} \right] \tag{8}$$

$$a^{ij}_{weight} = \frac{exp(a^{ij}_{score})}{\sum_{l\in\mathcal{N}_i} exp(a^{ij}_{score})} \tag{9}$$

$$A_{SEN} = H_{SEN} + \sum_{j\in\mathcal{N}_i} a^{ij}_{weight}[W^j_V \cdot H^j_{SEN}] \tag{10}$$

where [;] is the concatenated representation of two vectors; and W_a, W^i_Q, W^j_K, W^j_V are trainable weights.

The agent-like attention feature fusion model in this study employs a calculation method similar to that of agent attention. Moreover, the standard attention mechanism is used to calculate the agent, thereby aggregating semantic information from both the relation vector and sentence vector in the relation extraction task.

Q and K are used to perform softmax attention weight calculations on the agent. This process can be viewed as refining the sentence and relation vectors to retain effective semantic information, thereby completing the aggregation of information for linear attention calculations.

Next, we detail the process of fusing agent-like attention features (see Fig. 7).

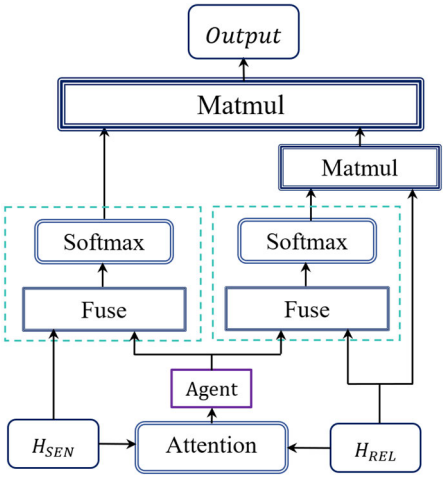

FIGURE 7. Process of fusing agent-like attention features.

First, to enhance the ability of the model to express semantic information of attention parameters, we employ the same procedure used to calculate agent parameter A_{SEN}. This involves sending agent parameter A_{SEN}, sentence vector H_{SEN}, and relationship category vector H_{REL} to the linear layer to obtain the respective weight parameters. Second, the model calculates the attention weights between the agent and sentence vector H_{SEN} and relationship category vector H_{REL}, respectively. During the feature fusion process, H_{SEN} and H_{REL} are concatenated with the agent vector and sent to two separate linear layers for the respective calculation of attention scores. Third, the softmax function is applied to calculate the attention weight for the two scores, as follows:

$$a^{ij\_w}_{REL\_A} = softmax\left(W_{ra}\left[W^i_R \cdot H^i_{REL}; W^j_A \cdot Agent^j\right]\right) \tag{11}$$

$$a^{kj\_w}_{SEN\_A} = softmax\left(W_{sa}\left[W^k_S \cdot H^k_{SEN}; W^j_A \cdot Agent^j\right]\right) \tag{12}$$

where [;] refers to the concatenated representation of two vectors; $W_{ra}, W_{sa}, W^i_R, W^k_S$, and W^j_A are trainable weights; *and* $softmax(\cdot)$ refers to the softmax function.

Finally, the model obtains the final attention output vector O_{REL} by calculating H_{SEN} using linear attention based on the

two attention weights, $a^{ij\_w}_{REL\_A}$ and $a^{kj\_w}_{SEN\_A}$, as follows:

$$O_{REL} = H_{REL} + \sum_{j \in \mathcal{N}_i} a^{ij\_w}_{SEN\_A} \sum_{I \in \mathcal{N}_i} a^{ij\_w}_{REL\_A} [W_V \cdot H_{REL_j}] \tag{13}$$

where [;] refers to the concatenated representation of two vectors; $W_{ra}, W_{sa}, W^i_R, W^k_S, W^j_A$ are trainable weights; and $\sigma(\cdot)$ refers to the softmax function.

C. SEQUENCE ENTITY TAGGER

The fusing of features in the agent-like attention layer results in two vectors rich in relational semantic information, including agent parameter A_{SEN} and final output O_{REL}. These vectors are then used by the sequence entity tagger to extract the subject and relation-specific object in the relation extraction task.

In the previous chapter, we introduced the method used in subject- and object-specific entity tagging. Note that the model uses sentence vector H_{SEN} as the value parameter to calculate the agent in order to obtain agent parameter A_{SEN}. This implies that after feature fusion and updating, A_{SEN} can be considered a context vector containing semantic information about the relation.

A_{SEN} is the context sentence vector updated by H_{SEN}, where A_{SEN} and H_{SEN} share the same sequence lengths. This implies that each token vector in the A_{SEN} sequence is richer in relation information than is H_{SEN}. Thus, we use A_{SEN} as the input vector for subject tagging and sequence labeling to predict the entity boundary of each token in A_{SEN}. The model performs a separate binary classification prediction for each token and then determines whether the token represents the start or end position of a subject.

Fig. 8 illustrates the process by which the subject tagger extracts the subject entity using updated context vector A_{SEN}. Each token in the A_{SEN} sequence is sequentially sent to the fully connected layer, and each token vector is weighted by the fully connected layer to generate an output value, which is passed through a Sigmoid function to limit the output range to 0∼1, indicating the probability that the token represents the start or end position of the entity. In accordance with the preset threshold, a token in a sequence with an output probability exceeding the threshold is labeled as 1, whereas a token with a probability less than the threshold is labeled as 0.

The model implements the entity extraction process of A_{SEN} by generating two output sequences (subject start-tag and subject end-tag), respectively representing the start and end positions of all subjects in the sentence.

Finally, we construct a span of the subject entity based on each token marked as 1 in the subject start-tag sequence, and then locate the nearest token marked as 1 in the subject end-tag sequence, and repeat the process until the subject of all spans in the original sentence have been extracted, as follows:

$$P^i_{start\_s} = \sigma\left(w^s_{start} \cdot A^i_{SEN} + b^s_{start}\right) \tag{14}$$

$$P^i_{end\_s} = \sigma\left(w^s_{end} \cdot A^i_{SEN} + b^r_{end}\right) \tag{15}$$

where A^i_{SEN} refers to the i-th token in the input sequence; $P^i_{start\_s}$ and $P^i_{end\_s}$ respectively indicate the probability that a token represents the start and end position of a subject; w^s_{start} and w^s_{end} are trainable weights; b^r_{start} and b^r_{end} are biases; and $\sigma(\cdot)$ is the Sigmoid activation function.

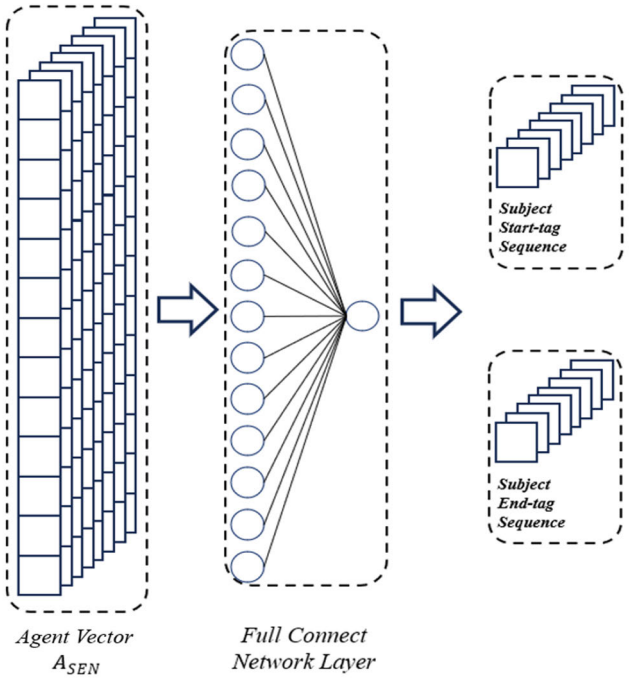

FIGURE 8. Process of subject entity extraction.

Following the extraction of all subject entities in the sentence, the next step is to find the object(s). Object extraction is similar to subject extraction, except that the extracted object must demonstrate a relation to the subject. Thus, the relation-specific object tagger adds semantic information related to the extracted subject to the original context vector. The token vector of each subject span extracted from the context sequence is then summed and averaged as a representation of subject-related semantic information. To ensure accuracy in linking objects and subjects according to predefined relationship categories, we also incorporate a relation category vector into the context vector. This vector is the final output O_{REL} from the agent-like attention feature fusion model. Next, we expand the three vectors (A_{SEN}, subject, and O_{REL}) to the same size and concatenate them before sending them to the fully connected layer for fusion. The output is then passed through a tanh function for normalization, as follows:

$$O^k_{SEN} = tanh\left(W_h\left[sub^k; A_{SEN}; O_{REL}\right] + b_h\right) \tag{16}$$

where O^k_{SEN} refers to the fuse representation vector that combines sub^k, A_{SEN}, and O_{REL}; sub^k is the representation vector of the k-th candidate subject; O_{REL} is the updated relation vector output in the agent-like attention layer ; *and* W_h and

b_h are trainable parameters in the fully connected network layer.

The extraction of objects according to category is based on the fused context representation vector O^k_{SEN}. Similar to the subject extraction process, after passing through the fully connected layer, the model predicts the probability for the boundary position of each token related to the object. When the predicted probability of the corresponding token exceeds the preset threshold, the corresponding position of the output sequence is labeled 1; otherwise, it is labeled 0. Finally, we output the subject start-tag sequence and subject end-tag sequence of the extracted objects under each specific relationship category to obtain the span of each relation-specific object. The extracted spans are then used to obtain the objects that correspond to the subject in all relations, yielding a relational triple, as follows:

$$P^i_{start\_o} = \sigma\left(w^r_{start} \cdot O^k_{SEN} + b^r_{start}\right) \tag{17}$$

$$P^i_{end\_o} = \sigma\left(w^r_{end} \cdot O^k_{SEN} + b^r_{end}\right) \tag{18}$$

where O^k_{SEN} refers to the fused representation vector; $P^i_{start\_o}$ and $P^i_{end\_o}$ indicate the probability that a token represents the start and end positions of an object; w^s_{start} and w^s_{end} are trainable weights, b^r_{start} and b^r_{end} are biases, and $\sigma(\cdot)$ is the Sigmoid activation function.

D. LOSS FUNCTION

In the entity extraction task, the sequence entity tagger confirms the boundary positions of the entities in the sentence, based on the predicted probability of each token. We established a target function for model training and optimization with the aim of enhancing the performance of the proposed model in the accurate extraction of all relational triples.

As outlined in Chapter 3, the computational model uses prediction probability for subject and object entity extraction, and the objective function calculates the probability of extracting relational triples. The model it trained to maximize the objective function in order to improve model performance.

To determine the probability of extracting relational triples, as calculated by the target function, we use the loss function to calculate the difference between the predicted probability and the truth. A cross-entropy function is used for the loss function, as follows:

$$\mathcal{L} = -\sum_{i=1}^{N} y_i \log\left(\hat{y}_i\right) \tag{19}$$

The goal of the relation extraction task is to maximize the probability of extracting relational triples. In Equation (4.19), y_i indicates the true probability result equal to 1, and $\hat{y}_i$ indicates the predicted probability of the model. We calculate $\mathcal{L}$ using the loss function to measure the gap between the prediction generated by the model versus the actual result. A low $\mathcal{L}$ value indicates that the prediction of their model is close to the actual result, indicating that the trained model is performing well.

Next, we perform a logarithmic transformation of the obtained objective function, converting the probability into logarithmic probability, which can then be used as the loss function. Note that the loss function for the relational triple is derived as follows:

$$\begin{aligned}\mathcal{L} &= \log \prod_{(s,r,o)\in T} P\left((s,r,o) \mid x\right) \\ &= \sum_{s\in T} \log P\left(s \mid x\right) + \sum_{r\in T|s} \log P\left(o \mid x,s,r\right) \\ &\quad + \sum_{r\in R\setminus T|s} logP\left(o_\phi \mid x,s,r\right)\end{aligned} \tag{20}$$

Finally, we use stochastic gradient descent (SGD) to maximize the loss function $\mathcal{L}$ in training the model.

V. EXPERIMENTS

In this section, we assess the efficacy of our agent-like attention feature fusion model in performing joint-entity relation extraction tasks when applied to two public datasets. We first outline the datasets used for relation extraction and evaluation metrics. We then examine the baseline methods used as a reference for comparisons with the proposed method.

A. DATASET

The model was assessed using the NYT [20] and WebNLG [21] public datasets.

1) NYT DATASET

The NYT dataset was developed by Riedel et al. for relation extraction tasks [20] in 2010. This dataset contains relation entity pairs from the New York Times (NYT) corpus with automated annotation provided through the freebase knowledge base to obtain complete relation triple data. The dataset is divided into a training set (56,195 training sentences) and a test set (500 test sentences), covering 24 relation types.

2) WebNLG DATASET

The WebNLG dataset was developed by Gardent et al. [21] in 2017 as a benchmark dataset for natural language generation (NLG). The dataset was generated by converting structured data from the knowledge graph of DBpedia into natural language text. Thus, this dataset comprises mainly structured data and corresponding natural language text, where the structured data represents the relationship or attribute information between entities.

When performing the relation extraction task, we organize the structured data into the format of relational triples, namely (*subject*, *relation*, *object*). Natural language text is used to train and test the model, comparing the results with actual relation triples to evaluate the model in terms of extraction performance. The WebNLG dataset includes 5019 training sentences and 703 test sentences, covering 171 categories.

The performance of the proposed model in relational triple extraction was assessed by classifying the data of overlapping triple categories within the two datasets. The extraction performance was then verified for three types of overlapping triple via experiments on the two complete datasets. Table 1 lists the quantities of information related to each category in the two datasets.

TABLE 1. Numbers of triples in each dataset.

Dataset / Category	NYT		WebNLG	
	Train	Test	Train	Test
Normal	37013	3266	1596	246
EPO	9782	978	227	26
SEO	14735	1297	3406	457
ALL	56195	5000	5019	703

TABLE 2. Form of confusion matrix.

Confusion Matrix		Ground Truth	
		Positive	Negative
Prediction Results	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

B. EVALUATION METRICS

Table 2 presents the confusion matrix [22] used to evaluate the performance of the model in relation extraction. The confusion matrix summarizes the prediction results by category, allowing us to count correct and incorrect predictions. The rows represent the true category of the data, while the columns represent the predicted category. Correct classifications are labeled as positive, and incorrect classifications are labeled as negative. The confusion matrix breaks down the prediction results as follows: true positive (TP), true negative (TN), false positive (FP), and false negative (FN).

The relation extraction performance of the proposed model and existing models was assessed in terms of precision, recall, and F1-score.

Precision refers to the proportion of correct predictions among all samples that were labeled positive in the prediction result field. Precision is derived as follows:

$$precision = \frac{TP}{TP + FP} \tag{21}$$

Recall refers to the proportion of correct predictions among all samples that were actually positive in the ground truth field. Recall was derived as follows:

$$recall = \frac{TP}{TP + FN} \tag{22}$$

The F1-score is a harmonic mean used to provide a comprehensively indication of performance by balancing precision and recall. The F-1 score was derived as follows:

$$\text{F1} - score = 2 \times \frac{precision \times recall}{precision + recall} \tag{23}$$

C. EXPERIMENTAL RESULTS

1) COMPARISONS

In designing the proposed relation extraction model, we implemented the training strategy proposed by Wei et al. [3]. We then compared the effectiveness of our models with the following methods: NovelTagging [4], CopyRe [23], GraphRel [24], TPLinker [5], DualDec [25], and CasRel [3].

2) MAIN RESULTS

Table 3 compares the relational triple extraction performance in experiments involving the NYT and WebNLG datasets. Note that data pertaining to the other models was sourced from their respective original papers, and the label "*" indicates situations where the experiment had to be rerun.

TABLE 3. Main results: NYT and WebNLG datasets.

Method	NYT			WebNLG		
	Prec.	Rec.	F1	Prec.	Rec.	F1
NovelTagging	62.4	31.7	42.0	52.5	19.3	28.3
CopyRe	61.0	56.6	58.7	37.7	36.4	37.1
GraphRel	63.9	60.0	61.9	44.7	41.1	42.9
TPLinker	91.3	92.5	91.9	91.8	92.0	91.9
DualDec	90.2	90.9	90.5	90.3	91.5	90.9
CasRel	89.7	89.5	89.6	93.4	90.1	91.8
CasRel*	88.3	90.2	89.2	90.9	90.4	90.7
AARel(Ours)	92.2	91.8	92.0	93.1	92.1	92.6

As shown in Table 3, the proportion of overlapping triples was higher when the models were applied to the NYT dataset than when applied to WebNLG. This means that the NYT dataset was more difficult to process. However, experimental results show that compared with CasRel*, we observed a more pronounced improvement in F1-score when applied to the NYT dataset than to WebNLG dataset. The F1-scores revealed that the inclusion of CasRel* improved performance on the two datasets by 2.8% and 1.9%, respectively. The proposed model consistently outperformed the other methods.

3) EXTRACTION RESULTS AND OVERLAPPING TRIPLES

The relational triples were divided into Normal, *SingleEntityOverlap* (SEO), and *EntityPairOverlap* (EPO), and test subsets were derived from the two test datasets accordingly. Table 1 lists the number of sentences in each test subset. Note that some sentences contain both *SingleEntityOverlap* (SEO) and *EntityPairOverlap* (EPO) relational triples, such that the total number of sentences in the three test subsets does not equal the total number of sentences in the test set.

Again, we used the F1-score to indicate the overall performance of the models in extracting overlapping relational

TABLE 4. Comparison of models in dealing with overlapping triples when applied to the NYT and WebNLG datasets (F1-score).

Model \ Dataset	NYT			WebNLG		
	Normal	SEO	EPO	Normal	SEO	EPO
CopyRe	66.0	48.6	55.0	59.2	33	36.6
GraphRel	69.6	51.2	58.2	38.3	40.6	66
TPLinker	90.1	93.4	94.0	87.9	92.5	95.3
DualDec	88.2	92.8	92.9	86.2	88.9	88.5
CasRel	87.3	91.4	92.0	89.4	94.7	92.2
AARel(Ours)	90.2	93.7	93.6	90.7	92.4	96.6

triples across the three categories. As shown in Table 4, the proposed method outperformed CasRel in the extraction of overlapping triples in all categories.

4) EXTRACTION RESULTS AND THE NUMBER OF RELATION TRIPLES

We also assessed the extraction ability of the models as a function of scenario complexity (i.e., different numbers of relation triples).

Similar to the processing of overlapping triples, we divided the public test sets into five subsets of $N = 1$, $N = 2$, $N = 3$, $N = 4$, and $N \geq 5$, where N indicates the number of relation triples in the target sentence. Table 5 lists the number of sentences in each subset.

TABLE 5. Number of relation triples in each data subset E).

Dataset	$N = 1$	$N = 2$	$N = 3$	$N = 4$	$N \geq 5$
NYT	3244	1045	312	291	108
WebNLG	266	171	131	90	45

TABLE 6. Comparison of extraction models as a function of the numbers of triples (F1-score).

Model \ Dataset	NYT					WebNLG				
	$N = 1$	$N = 2$	$N = 3$	$N = 4$	$N \geq 5$	$N = 1$	$N = 2$	$N = 3$	$N = 4$	$N \geq 5$
CopyRe	67.1	58.6	52.0	53.6	30.0	59.2	42.5	31.7	24.2	30.0
GraphRel	71.0	61.5	57.4	55.1	41.1	66.0	48.3	37.0	32.1	32.1
TPLinker	90.0	92.8	93.1	96.1	90.0	88.0	90.1	94.6	93.3	91.6
CasRel	88.2	90.3	91.9	94.2	83.7	89.3	90.8	94.2	92.4	90.9
AARel (Ours)	90.2	92.3	92.5	95.5	88.1	90.9	91.9	94.7	93.1	92.5

We then used the F1-score to assess the performance of the model when applied to sentences with multiple triples, the results of which are shown in Table 6. Overall, the proposed model outperformed CasRel when applied to complex scenarios involving sentences with multiple relation triples.

TABLE 7. Training time of extraction models (sec/epoch).

Method	Training Time (sec.)	
	NYT	WebNLG
TPLinker	1100	331
CasRel	426	43
AARel(Ours)	514	61

5) TRAINING TIME

The experiment results demonstrate the excellent relation extraction performance of the proposed agent-like feature fusion model. AARel slightly outperformed TPLinker in the initial analysis; however, TPLinker proved more effective when applied to complex scenarios. Table 7 lists the training time results indicting the time required to run one training epoch when applied to the two public datasets. Overall, the training time of the proposed model was on par with that of CasRel, both of which far outperformed TPLinker.

VI. CONCLUSION

This paper introduces an agent-like model for the fusion of attention features in relation extraction. The agent attention mechanism was meant to improve upon the CasRel model, which is based on a cascade binary annotation method. The proposed model employs a novel attention module built upon the original cascaded binary tagging framework to enhance efficiency in extracting relational entities. The agent-like fusion of attention features enables the extraction of implicit semantic information within relational entities and relationship categories by training an agent vector that encompasses information related to both the context and relationship category. In experiments, the proposed model significantly outperformed CasRel in terms of relation extraction performance and the accurate extraction of relational triples.

REFERENCES

[1] M. A. Hearst, "Automatic acquisition of hyponyms from large text corpora," in *Proc. 14th Conf. Comput. Linguistics*, vol. 2, 1992, p. 539.

[2] Z. Zhong and D. Chen, "A frustratingly easy approach for entity and relation extraction," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2021, pp. 50–61.

[3] Z. Wei, J. Su, Y. Wang, Y. Tian, and Y. Chang, "A novel cascade binary tagging framework for relational triple extraction," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 1476–1488.

[4] S. Zheng, F. Wang, H. Bao, Y. Hao, P. Zhou, and B. Xu, "Joint extraction of entities and relations based on a novel tagging scheme," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2017, pp. 1227–1236.

[5] Y. Wang, B. Yu, Y. Zhang, T. Liu, H. Zhu, and L. Sun, "TPLinker: Single-stage joint extraction of entities and relations through token pair linking," in *Proc. 28th Int. Conf. Comput. Linguistics*, 2020, pp. 1572–1582.

[6] Y. Wang, C. Sun, Y. Wu, H. Zhou, L. Li, and J. Yan, "UniRE: A unified label space for entity relation extraction," in *Proc. 59th Annu. Meeting Assoc. Comput. Linguistics 11th Int. Joint Conf. Natural Lang. Process.*, vol. 1, 2021, pp. 220–231.

[7] C. Sutton, K. Rohanimanesh, and A. McCallum, "Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data," in *Proc. 21st Int. Conf. Mach. Learn. (ICML)*, 2004, p. 99.

[8] C. Cortes and V. Vapnik, "V. support-vector networks," *Mach. Learn.*, vol. 20, pp. 273–297, Sep. 1995.
[9] D. Zelenko, C. Aone, and A. Richardella, "Kernel methods for relation extraction," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, vol. 10, 2002, pp. 71–78.
[10] M. Miwa and M. Bansal, "End-to-end relation extraction using LSTMs on sequences and tree structures," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2016, pp. 1105–1116.
[11] Y. Lin, S. Shen, Z. Liu, H. Luan, and M. Sun, "Neural relation extraction with selective attention over instances," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2016, pp. 2124–2133.
[12] C. Alt, M. Hübner, and L. Hennig, "Improving relation extraction by pre-trained language representations," 2019, *arXiv:1906.03088*.
[13] D. Ye, Y. Lin, P. Li, and M. Sun, "Packed levitated marker for entity and relation extraction," in *Proc. 60th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2022, pp. 4904–4917.
[14] J. Zhao, W. Zhan, X. Zhao, Q. Zhang, T. Gui, Z. Wei, J. Wang, M. Peng, and M. Sun, "RE-matching: A fine-grained semantic matching method for zero-shot relation extraction," in *Proc. 61st Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2023, pp. 6680–6691.
[15] R. Zhang, Y. Li, and L. Zou, "A novel table-to-graph generation approach for document-level joint entity and relation extraction," in *Proc. 61st Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2023, pp. 10853–10865.
[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Neural Inf. Process. Syst.*, 2017, pp. 1–11.
[17] D. Han, T. Ye, Y. Han, Z. Xia, S. Pan, P. Wan, S. Song, and G. Huang, "Agent attention: On the integration of softmax and linear attention," 2023, *arXiv:2312.08874*.
[18] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. North Amer. Chapter Assoc. Comput. Linguistics*, 2019, pp. 1–16.
[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, vol. 16, 2016, pp. 770–778.
[20] S. Riedel, L. Yao, and A. McCallum, "Modeling relations and their mentions without labeled text," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, Sep. 2010, pp. 148–163.
[21] C. Gardent, A. Shimorina, S. Narayan, and L. Perez-Beltrachini, "Creating training corpora for NLG micro-planners," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2017, pp. 179–188.
[22] S. V. Stehman, "Selecting and interpreting measures of thematic classification accuracy," *Remote Sens. Environ.*, vol. 62, no. 1, pp. 77–89, Oct. 1997.
[23] X. Zeng, D. Zeng, S. He, K. Liu, and J. Zhao, "Extracting relational facts by an end-to-end neural model with copy mechanism," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2018, pp. 506–514.
[24] T.-J. Fu, P.-H. Li, and W.-Y. Ma, "GraphRel: Modeling text as relational graphs for joint entity and relation extraction," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 1409–1418.
[25] J. Cheng, T. Zhang, S. Zhang, H. Ren, G. Yu, X. Zhang, S. Gao, and L. Ma, "A cascade dual-decoder model for joint entity and relation extraction," *IEEE Trans. Emerg. Topics Comput. Intell.*, early access, Jun. 6, 2024, doi: 10.1109/TETCI.2024.3406440.

JIM-WEI WU (Member, IEEE) received the B.S. and M.S. degrees from National Taiwan Normal University, Taipei City, Taiwan, in 2005 and 2007, respectively, and the Ph.D. degree in electrical engineering from National Taiwan University, Taipei City, in 2013.

He is currently an Associate Professor with the Department of Electrical Engineering and the Deputy Director with the Center for Academia and Industry Collaboration, National Central University, Taoyuan, Taiwan. His research interests include text mining, deep learning theory, robotic systems, visual technology, precision positioning control, micro-/nano-measurement systems, and advanced control theory and applications. He is an Associate Editor of IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT.

HANG-KAI YE was born in Ningbo, China, in 1999. He received the B.S. degree in automatic engineering from Feng Chia University, Taichung City, in 2022, and the M.S. degree in electrical engineering from National Central University, Taoyuan, Taiwan, in 2024.

His research interests include text mining, natural language processing, relation extraction, and deep learning theory and applications.

JIA-CHENG LI was born in New Taipei City, Taiwan, in 2001. He received the B.S. degree in mechanical engineering from National Central University, Taoyuan, Taiwan, in 2023. He is currently pursuing the M.S. degree in electrical engineering with National Central University, Taoyuan. His research interests include image processing, 3D reconstruction, and deep learning theory and applications.

JUNG-YU LIAO received the Ph.D. degree in health promotion and health education from National Taiwan Normal University, Taipei, Taiwan, in 2019.

She is currently a Faculty Member and an Assistant Professor with the Department of Health Promotion and Health Education, College of Education, National Taiwan Normal University, Taipei. Her research interests include text mining, GenAI application, social and behavioral health sciences, gerontological health promotion, and substance abuse prevention.