N-ary Cross-sentence Relation Extraction: From Supervised to Unsupervised Learning

Chenhan Yuan

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science and Application

Hoda Eldardiry, Chair
Lifu Huang
Ismini Lourentzou

April 26, 2021
Blacksburg, Virginia

Keywords: Relation Extraction, Unsupervised Learning, Autoencoder, Reinforcement Learning.

N-ary Cross-sentence Relation Extraction: From Supervised to Unsupervised Learning

Chenhan Yuan

(ABSTRACT)

Relation extraction is the problem of extracting relations between entities described in text. Relations identify a common "fact" described by distinct entities. Conventional relation extraction approaches focus on supervised binary intra-sentence relations, where the assumption is relations only exist between two entities within the same sentence. These approaches have two key limitations. First, binary intra-sentence relation extraction methods can not extract a relation in a fact that is described by more than two entities. Second, these methods cannot extract relations that span more than one sentence, which commonly occurs as the number of entities increases. Third, these methods assume a supervised setting and are therefore not able to extract relations in the absence of sufficient labeled data for training. This work aims to overcome these limitations by developing n-ary cross-sentence relation extraction methods for both supervised and unsupervised settings. Our work has three main goals and contributions: (1) two *unsupervised* binary intra-sentence relation extraction methods, (2) a *supervised n-ary cross-sentence* relation extraction method, and (3) an *unsupervised n-ary cross-sentence* relation extraction method. To achieve these goals, our work includes the following contributions: (1) an automatic labeling method for n-ary cross-sentence data, which is essential for model training, (2) a reinforcement learning-based sentence distribution estimator to minimize the impact of noise on model training, (3) a generative clustering-based technique for intra-sentence unsupervised relation extraction, (4) a variational autoencoder-based technique for unsupervised n-ary cross-sentence relation

extraction, and (5) a sentence group selector that identifies groups of sentences that form relations.

N-ary Cross-sentence Relation Extraction: From Supervised to Unsupervised Learning

Chenhan Yuan

(GENERAL AUDIENCE ABSTRACT)

In this work, we designed multiple models to automatically extract relations from text. These relations represent the semantic connection between two or more proper nouns. Previous work include models that can only extract relations between two proper nouns in a single sentence, while the methods proposed in this thesis can extract relations between two or more proper nouns in multiple sentences. We propose three models. The first model can automatically remove erroneous annotations in training data, thereby making the models more credible. We also propose a more effective model that can automatically extract relations between two proper nouns in a single sentence without the need for data annotation. We later extend this model so that it can extract relations between two or more proper nouns in multiple sentences.

Acknowledgments

I am deeply grateful to my advisor Professor Hoda Eldardiry. Her continuous support and invaluable advice encouraged me to overcome difficulties in research and daily life during my graduate study.

I also would like to thank Professor Ismini Lourentzou and Professor Lifu Huang for their insightful comments and suggestions. It is their kind help that made this thesis better.

My appreciation also goes out to my family back home and my friends for their encouragement and support all through my studies.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

A Knowledge Graph (KG) is a popular knowledge data arrangement system that has been deployed in many common artificial intelligence (AI) tasks. This includes search engines, recommender systems, and question answering [57, 58, 65]. Currently, a KG is composed of triplets consisting of (head entity, relation, tail entity). However, in practice, some relations contain more than two entities. For example, the relation "educate" in WikiData [56] includes four entities, the person's "name", "academic degree", "academic major" and "school". Some common KGs consider these relations involving multiple entities and the associated n-ary relations. These KGs include Freebase and WikiData [5, 56]. However, complex n-ary relations that connect multiple entities have mainly been extracted manually from the text, which is time-consuming and expensive.

As a key step in automatic knowledge graph construction, relation extraction is the task of extracting the relation between entities expressed in a sentence. Previous work has largely focused on intra-sentence binary relation extraction, where the goal is to extract the relation between an entity pair in a sentence [49]. The output of relation extraction models are the triplets that are used in KG construction. In order to automatically extract n-ary relations from raw text, a recently proposed research task is defined as n-ary cross-sentence relation extraction [37]. The current research in this area mainly focuses on the design of supervised n-

ary cross-sentence relation extraction models based on distant supervision. However, distant supervision assumes that if two entities that appear in a sentence also appear in a KG, then this sentence describes the relation in the KG between the two entities. Therefore, distant supervision introduces a lot of noise to the dataset, which results in unreliable model training results [46]. In addition, there is currently available manually labeled high-quality datasets for this task, which also poses challenges to the design of such models.

Due to the two aforementioned reasons, in this work, our goal is to reduce the impact of noise data and design a model that can utilize unlabeled data. We first proposed a reinforcement learning-based estimator model, which can remove the noise in the dataset. This ensures the reliability of the performance of the trained relation extractor. We further proposed two new unsupervised binary intra-sentence relation extraction models; the extension of which are used as the architecture of the unsupervised n-ary cross-sentence relation extraction model. This unsupervised model is iteratively trained with a selector, which can determine whether the input sentence group does have n-ary relation information.

1.2 Contributions

The contributions of this work and the corresponding chapters can be summarized as follows:

- A reinforcement learning-based sentence distribution estimator is proposed to remove the impact of noise in distant supervision labeled dataset in Chapter 3.
- An n-ary cross-sentence relation extractor is proposed to encode both consecutive and non-consecutive sentences in Chapter 3.
- An autoencoder-based unsupervised binary intra-sentence relation extraction model is proposed to train an encoder that can output relation information given the input

sentence in Chapter 4.

- A variational autoencoder-based probabilistic model is proposed to train a relation classifier without labeling information in Chapter 4.

- A selection-guided variational antuencoder-based unsupervised n-ary cross-sentence relation extraction model is proposed in Chapter 5. The selector component of this model can determine which input sentence group has n-ary relation information. Then the VAE architecture can train any supervised n- ary cross-sentence relation extraction models by jointly training with the decoder reconstruction process.

Other chapters of this thesis are organized as follows: the background techniques used in n-ary cross-sentence relation extraction are briefly introduced in Chapter 2. Chapter 6 draws the conclusion of this work.

Chapter 2

Background

2.1 Deep Learning for Relation Extraction

2.1.1 Convolutional Neural Networks

Convolutional Neural Network (CNN) was proposed to solve the image recognition problem. CNN uses multiple convolutional filters to extract locally sensitive information in the image separately, and integrates the information extracted by multiple filters through pooling in the output stage [25]. Extensively experiments demonstrate that CNN can extract spatial hierarchies of the input signal features [59].

CNN has also been used in the field of text extraction. Currently in the binary intra-sentence relation extraction model, the most commonly-used CNN variant is Piece-wise CNN (PCNN). The core idea of PCNN is to divide the input sentence into three segments according to the positions of the two entities in the sentence. Then convolutional filters perform convolution calculations on these three segments separately, and finally merge the calculation results [63]. The reason why PCNN can work is that manually segmenting the input sentence is similar to telling filters in advance what is the locally important information in the input, thus this process can speed up the convergence of the model.

2.1.2 Attention Mechanism

Attention mechanism was first introduced in neural machine translation tasks. When the model translates a certain word, the attention mechanism enables the model to calculate the weight of the impact of each token in the input sentence on the current translated word [28]. At present, the most commonly used attention mechanisms are soft attention and multi-head attention [28, 54]. Soft attention defines the influence of the feature vector of each token on the current output by learning the similarity between the feature vector of each input token and a predefined query vector. The mathematical definition of one simple form soft attention can be formulated as follows:

$$\begin{aligned} p_k &= \sum_{j=0}^{m} \epsilon_{k,j} u_j \\ \epsilon_{k,j} &= \frac{e^{c_{k,j}}}{\sum_{i=0}^{m} e^{c_{k,i}}} \\ c_{k,j} &= u_j^T r_k \end{aligned} \tag{2.1}$$

where r_k is the k-th query vector and u_j is the feature vector of i-th input token.

Self-attention expands this definition. In self-attention, each token corresponds to three feature vectors, which are query vector, key vector, and value vector [54]. The model learns the similarity between the feature vectors of each token. Experiments show that most NLP tasks using multi-head attention can achieve better performance than that of using various sequential neural network models [13].

2.1.3 Recurrent Neural Networks

Recurrent Neural Networks can model sequential data and hence have been widely used in NLP and audio sequence labeling tasks [6, 16, 27, 32]. RNN has three components, the input

state, output state, and the hidden state. The core idea of RNN is that the hidden state of time step t is affected by both the input state at time step t and the hidden state of the previous time step $t-1$. However, this mechanism can propagate gradient exponentially and causes some problems when encoding long sequence. For example, when the input sentence is too long, RNN encounters problems such as vanishing gradient or gradient explosion [36]. In order to solve this problem, some researchers proposed a variant of RNN, which is long-short term memory (LSTM) [22]. This structure enables the model proactively choose to forget some previous information to avoid gradient-related problems. The core idea of this model is to add a "forget gate" to the hidden state of the RNN, which controls which information will be passed on. Following this work, some researchers argue that the appearance of a word or phrase in the text should be affected not only by the semantic information before it, but also by the subsequent semantic information. That is, context is important for words prediction. Therefore, Bidirectional LSTM is proposed to model context information [15].

2.2 Learning Approach

2.2.1 Distant Supervision

Distant supervision is an automatic dataset annotation method [34]. Distant supervision assumes that if two entities appearing in a sentence also appear in a Knowledge Graph, then this sentence describes the relation saved in the KG between the two entities. This automatic labeling method is only suitable for binary intra-sentence relation extraction task. Therefore, some researchers expanded the definition of distant supervision such that this approach can be applied on n-ary cross-sentence relation extraction task. The definition is that if the consecutive sentences (a sentence group) contain the entities that have a relation

in a knowledge base, these sentences as a whole describe that relation [41]. However, this definition assumes that n-ary relation can only appear in consecutive sentences, which is not always true. Therefore, we propose to further relax the previous assumption in Chapter 3.

2.2.2 Self-supervised Learning

Self-supervised learning uses part of the information in the unlabeled dataset as the labels such as the prediction of masked word in the given sentence. Then the input of supervised learning models is the remaining information while the prediction of these models are the labels. In this way, the supervised learning model can be trained with an unlabeled dataset [20]. In relation extraction task, self-supervised learning is achieved by applying variational autoencoder models [30]. In this method, the decoder reconstructs the triplet in the input sentence according to the probability of the encoder output. In order to allow the encoder output a variety of relations, an entropy-based regularization function is added to the original objective function. In this way, optimizing the model is similar to optimizing a variational autoencoder. More recently, some work considered that this model is not stable [48]. This is because the model only predicts one same relation for all input sentences, and the predicted relations probability distribution for each instance is similar. This recent work proposed that in addition to the original entropy-based regularization, dispersion loss needs to be added, which is the KL divergence of relation distribution and uniform distribution. This loss term requires that all relation classification results of the model conform to the uniform distribution. That is, every relation has a chance to be the predicted result of the model. They also applied the Piece-wise Convolutaional Neural Network (PCNN) as the encoder architecture instead of feature selection procedure in previous unsupervised relation extraction works. The PCNN architecture applied in this model is then replaced by entity type feature, since entity type was considered as the most important feature [53].

2.2.3 Reinforcement Learning

An intelligent agent is trained with Reinforcement learning (RL) by interacting with the environment. The parameters of this agent are updated by maximizing the cumulative reward the agent received during the interaction [51]. There are three components in one RL agent, which are $\{a, s, \pi\}$ where a is the action set that the agent may choose and s is the state set of the agent and π is the policy function, the agent uses which to make next choice. In n-ary cross-sentence relation extraction task, RL approach is applied to automatically select the high-quality distant supervision labeled sentence group. For example, in binary intra-sentence relation extraction task, some work trained an extra selector model, which selected the correctly labeled sentences as the training data of the relation extraction model [12, 40, 61]. Most selectors are reinforcement learning (RL)-based models. These selectors are iteratively trained with the relation extraction models such that two models are well-trained after training stage.

2.2.4 Unsupervised Learning

Unsupervised Learning is a machine learning technique that learns patterns from unlabeled data [2]. Unsupervised relation extraction is a way to cluster entity pairs with the same relations and label the cluster automatically or manually. Hasegawa et al. first proposed the concept of the context of entity pairs, which can be deemed as extracted features from sentences. After that, they clustered different relations based on feature similarity and selected common words in the context of all entity pairs to describe each relation [18]. Following this work, an extra unsupervised feature selection process was proposed to reduce the impact of noisy words in context [8]. Yan et al. proposed a two-step clustering algorithm to classify relations, which included a linguistic patterns based cluster and a surface context

cluster. The linguistic patterns here are pre-defined rules derived from the dependency tree [60]. Poon and Domingos also thought of using dependency trees to cluster relations. The dependency trees are first transformed to quasi-logical forms, where lambda forms can be induced recursively [39]. Rosenfeld and Feldman, on the other hand, considered that arguments and keywords are relation patterns that can be learned by utilizing instances [45]. Their approach was an improvement of KnowItAll system, which is a fact extraction system focusing more on entity extraction [11].

Some works also considered unsupervised relation extraction as a probabilistic generation task. Latent Dirichlet Allocation (LDA) was applied in unsupervised relation extraction [4, 62]. Researchers replaced the topic distributions with triplets distributions and implemented Expectation Maximization algorithm to cluster similar relations. de Lacalle and Lapata applied this method in general domain knowledge, where they first encoded a Knowledge Base using First Order Logic rules and then combined this with LDA [9]. In unsupervised open domain relation extraction [10], the authors used corresponding sentences of entity pairs as features and then vectorized the features to evaluate similarity of relations. These features include the re-weighting word embedding vectors and types of entities.

2.3 Related Work on N-ary Cross-sentence Relation Extraction

Dependency shortest path has been applied with other pre-processing features for n-ary cross-sentence relation extraction [26, 31]. With the rise of deep learning, some work encoded the dependency shortest path via graph neural networks. Peng et al. applied Graph-LSTM to encode the dependency shortest path and link each path [37]. One dependency shortest path

usually requires two Graph-LSTMs. Song et al. proposed the Graph-state LSTM so that only one Graph-LSTM is needed to encode a path [50]. Some work also implemented Bi-LSTM directly to encode the whole sentence sequences without requiring any preprocessing [29]. The LSTM-CNN model they proposed achieved a better performance on the PubMed dataset, but it cannot encode long sequences. Recently, this model has been improved by deploying a multi-head attention layer. The model is also enhanced by incorporating prior knowledge from a pre-trained Knowledge Base [66].

Chapter 3

Supervised N-ary Cross-sentence Relation Extraction

3.1 Problem Formulation

In a relation extraction task, a fact is defined as a collection of i entities and one corresponding relation, where $i \geq 2$. The relations are verb phrases and describe the relationship among these entities. For every m sentences, a relation extraction model should give the relation among entities expressed in these sentences. If $m \geq 1$ and $i \geq 2$, the task is the cross-sentence n-ary relation extraction problem.

In distant supervision-based method, we decompose the cross-sentence n-ary relation extraction task into two sub-problems: sentence distribution estimation and relation extraction. The sentence distribution estimation is formulated as follows: given a set of sentence groups and relation label pairs as $\{(g_1, r_1), (g_2, r_2), \cdots, (g_n, r_n)\}$, where there are variable numbers of sentences in each sentence group g and r is the noisy relation label produced by distant supervision, the objective is to decide which sentences in each group truly describe the relation. In other words, the model tells which sentence is correctly labeled and should be selected as a training instance. The relation extraction is to classify relation r, given a sentence group g.

3.2 Proposed Relation Extractor

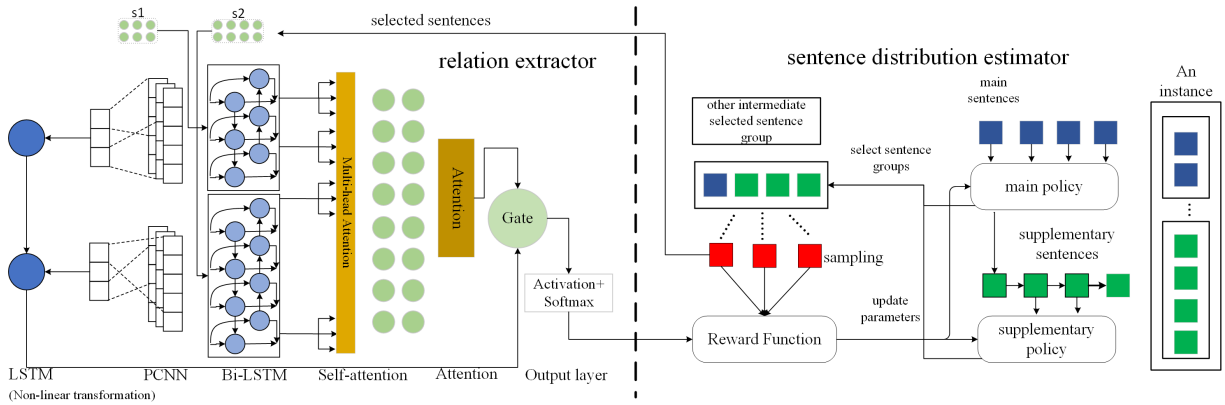

Figure 3.1: The flowchart of the proposed model. On the right side, the sentence distribution estimator consists of main policy and supplementary policy. The relation extraction model is on the left side.

As shown in Fig 3.1, the proposed model consists of two sub-models. The first is a sentence distribution estimator (SentDE), which is used to measure the probability of whether a sentence is correctly labeled by distant supervision. This model outputs a group of sentences that describe a complete fact. The second model is the relation extraction model (RE). This model takes a group of sentences as input and infers the relation contained in these sentences.

As shown in Fig 3.1, the relation extraction (RE) model receives a group of sentences from SentDE and then encodes them using a Bidirectional-LSTM layer. We implement attention and self-attention mechanisms to incorporate these sentence encodings to classify the relation. Conventional cross-sentence relation extraction models encode consecutive sentences so they do not consider the connection and context between sentences, which should be learned when encoding non-consecutive sentences. Therefore, attention mechanism is enriched with the output from the non-linear transformation (LSTM) layer by a gate layer in the proposed model. This layer learns how the information transforms in each sentence, which is exactly the context information. The input of this transformation layer is the Piecewise Convo-

lutional Neural Network (PCNN), which encodes each sentence as a feature vector. By applying the hybrid model of attention mechanism and non-linear transformation, the proposed model is universal for cross-sentence n-ary relation extraction task in many scenarios, including both non-consecutive and consecutive sentences.

3.2.1 Sentence Encoding

Bidirectional-LSTM is applied as the sentence encoding layer [14]. The input of Bi-LSTM is a sequence of the concatenation of the word embedding vector and position encoding vector. We pre-train the word embedding with d_w dimension by Glove [38]. The position encoding is implemented as follows. Supposing the ordered entity list of one sentence is $e_1 e_2 \dots e_n$, we calculate the position distances between words and e_1, e_n. Position distance is defined as the number of sentences between the two sentences in the text. Then these position distances are projected to a dense vector space, which has dimension d_p. Although we can select position distances of all entities, more position encoding features will decrease the classification accuracy [29]. The input dimension for Bi-LSTM of one sentence is $\mathbb{R}^{n_w \times (d_w + d_p + d_p)}$, where n_w is the number of words in each sentence. Then the output dimension from the Bi-LSTM of one sentence is $\mathbb{R}^{n_w \times d_b}$, where d_b is the hidden dimension of the Bi-LSTM.

3.2.2 PCNN and Non-linear Transformation Layer

The vector representation of each sentence is used in the non-linear transformation layer. We apply PCNN to take the output of Bi-LSTM layer and get the vector representation [63]. PCNN first uses n_f filters, each of kernel size $\mathbb{R}^{n_s \times d_b}$, to extract features, where n_s is the window size. The output of each filter f_i is then divided into three segments $\{f_{i1}, f_{i2}, f_{i3}\}$ according to entities e_1, e_n positions, then max-pooling is applied in each segment. Eq. 3.1

formally defines the piece-wise max-pooling layer:

$$\begin{aligned} p_{ij} &= max(f_{ij}), 1 \leq i \leq n_f, 1 \leq j \leq 3 \\ p_i &= p_{i1} \oplus p_{i2} \oplus p_{i3} \end{aligned} \tag{3.1}$$

where $\oplus$ denotes concatenation and $p_i \in \mathbb{R}^3$ is the piece-wise max pooling result of i-th filter. Then the output dimension of PCNN for one sentence is $\mathbb{R}^{3n_f}$.

The non-linear transformation layer is implemented with LSTM [22]. The sentence feature vector s_i coming from the PCNN layer is the input at each LSTM cell state. The hidden vector of the last state is the output of the non-linear transformation layer. The mathematical definition is shown in Eq. 3.2

$$\begin{aligned} h_i, c_i &= LSTM(s_i, h_{i-1}, c_{i-1}), 1 \leq i \leq n_{se} \\ h_0, c_0 &\sim \mathcal{N}(0, 1) \end{aligned} \tag{3.2}$$

where $\mathcal{N}(0, 1)$ denotes standard normal distribution and n_{se} is the number of sentences in the sentence group. $q \in \mathbb{R}^{d_h}$ is the output of the last hidden LSTM, where d_h is the hidden dimension of LSTM cell.

3.2.3 Attention and Self-attention Mechanism

Previous works report that multi-head self-attention improves sentence-level relation extraction performance because of its ability to model long sequences [54, 66]. This mechanism is

applied via Eq. 3.3:

$$
\begin{aligned}
M_i &= softmax\left(\frac{QW_i^Q(KW_i^K)^\top}{\sqrt{d}}\right)VW_i^V \\
M &= M_1 \oplus M_2 \oplus M_3 \oplus \cdots M_{n_{he}} \\
U &= MW^O
\end{aligned}
\tag{3.3}
$$

where $W_i^Q \in \mathbb{R}^{n_{se}\times\frac{d_s}{n_{he}}}$, $W_i^K \in \mathbb{R}^{n_{se}\times\frac{d_s}{n_{he}}}$, $W_i^V \in \mathbb{R}^{n_{se}\times\frac{d_s}{n_{he}}}$, $W_i^O \in \mathbb{R}^{d_s\times d_s}$ are learnable parameters and $Q \in \mathbb{R}^{n_{se}\times d_s}$,$K \in \mathbb{R}^{n_{se}\times d_s}$,$V \in \mathbb{R}^{n_{se}\times d_s}$ are the query, key and value vectors projected from the input vectors. n_{he} and d_s are the number of heads and the number of hidden units, respectively.

Another soft attention layer is applied to attend to the input U that contributes the most on the classification of the relation. As shown in Eq. 4.4, this layer compares the relation vectors with output vectors from multi-head self-attention.

$$
\begin{aligned}
p_k &= \sum_{j=0}^{m} \epsilon_{k,j} u_j \\
\epsilon_{k,j} &= \frac{e^{c_{k,j}}}{\sum_{i=0}^{m} e^{c_{k,i}}} \\
c_{k,j} &= u_j^T r_k
\end{aligned}
\tag{3.4}
$$

where $r_k \in \mathbb{R}^d$ is the learnable vector of k-th relation and $p_k \in \mathbb{R}^d$ is the attention result for relation r_k.

3.2.4 Gate Layer and Output Layer

As shown in Eq. 3.5, an element-wise gate layer is applied to incorporate the outputs from attention layer and non-linear transformation layer.

$$\begin{aligned} \alpha &= \sigma(W_a^\top p + b_a) \\ \tilde{S}_n &= tanh(W_n^\top q + b_n) \\ S &= \alpha p + (1-\alpha)\tilde{S}_n \end{aligned} \tag{3.5}$$

where $p \in \mathbb{R}^{d_s}$ is the attention result and $W_a \in \mathbb{R}^{d_s}$ is the weighting matrix. $q \in \mathbb{R}^{d_s}$ is the LSTM's result and $W_n \in \mathbb{R}^{d_s}$ is the weighting matrix.

3.3 Proposed Sentence Quality Estimator

The proposed model is a two-level agent reinforcement learning model as in Fig. 3.1. We assume that for each group of sentences, there is one main sentence and some supplementary sentences. The main sentence that has the two main entities may contribute the most about the relation. In other words, the relations cannot be extracted without the main sentence. The supplementary sentences supplement the relation information and are selected given the main sentence.

3.3.1 Main Sentence-Level Policy

State The vector representation of main sentence i is state s_i, and is generated by the PCNN layer of the RE model.

Action The action set for this level is $a_i \in \{0, 1\}$, where 1 indicates that the program selects

the sentence i as the correctly labeled sentence. Note that this is a one-state RL and the reward is calculated once a_i is decided.

Policy The policy π_θ represents the probability of selecting the input sentence given the encoding information s_i.

$$\begin{aligned} \pi_\theta(a_i, s_i) &= P(a_i|s_i) \\ &= \sigma(W^\top s_i + b) \end{aligned} \tag{3.6}$$

where σ is the sigmoid function and $W^\top \in \mathbb{R}^{d_s}$ is the weighting matrix and d_s is the dimension of the state vector.

Reward The reward is the classification accuracy of the relation extraction model, given selected input sentences.

3.3.2 Supplementary Sentence-Level Policy

State The state m_j of this level comprises three indicators and the encoding information of sentence c_j. The first indicator e^{-d} measures the distance between the current supplementary sentence and the main sentence, where d is the position distance and e is the exponential function. The second indicator $|\{e|e \in sent_j \wedge e \in E\}|$ gives the number of entities of the current sentence $sent_j$, where E is the set of entities of the fact. The third indicator $\frac{c_j \cdot s_i}{||c_j||_2 \cdot ||s_i||_2}$ measures the cosine similarity between the current sentence and the main sentence i. Along with the encoding information, we assume that these indicators can fully address the context information when selecting supplementary sentences.

This scenarios falls under a traditional finite RL setting. The transition function between each m_j should be defined to calculate the reward once the end state is reached. The transition function is defined as follows. We first sort the supplementary sentences descending according to their corresponding second indicators, then let the agent decide if selecting

the first sentence. The next state is the first sentence of the descending sorted remaining supplementary sentences according to $|\{e|e \in sent_j \wedge e \in E/E_{prev}\}|$, where E_{prev} is the entities in the previous selected sentences.

Action The action set for this level is $b_j \in \{0, 1\}$, where 1 indicates that the program selects the current sentence as the correctly labeled sentence, where the label is the relation indicated by the sentence.

Policy Eq. 3.7 shows the policy π_γ considers sentence-level indicators and sentence encoding information simultaneously.

$$\begin{aligned} \pi_\gamma(b_j, m_j) &= P(b_j|m_j) \\ &= \sigma\left(\alpha(W_k^\top k_j + b_k) + \beta(W_s^\top s_j + b_c)\right) \end{aligned} \tag{3.7}$$

where $W_k \in \mathbb{R}^3$ and $W_s \in \mathbb{R}^{d_s}$ are weighting matrices. k_j is the vector of three real-number indicators and d_s is the dimension of the encoding vector of the sentence. α and β are also learnable parameters.

Reward Note that the rewards, i.e., the accuracy of the results from the RE model, can only be calculated when all necessary sentences are given. Therefore, there is no intermediate reward that can be used directly for updating gradients. Similar to playing Go, we apply the Monte Carlo search algorithm to simulate possible future results and use the average of these results as the intermediate reward [47]. More formally, given the current state and previous states $m_{1:j}$, the Monte Carlo search algorithm with π_γ as roll-out policy is applied to sample the future possible state transitions $m'_{j+1:M}$, where M is the end state. The mathematical definition is in Eq. 3.8

$$Monte_{\pi_\gamma}(m_{1:j}; m'_{j+1:M}; N) = \{m_{1:j}m'^{(1)}_{j+1:M}, m_{1:j}m'^{(2)}_{j+1:M}, \ldots, m_{1:j}m'^{(n)}_{j+1:M}\} \tag{3.8}$$

where N is the number of samples. Based on this, the intermediate reward can be calculated via Eq. 3.9

$$R(i,j) = \begin{cases} \frac{1}{N}\sum_{n=1}^{N} e^{-\mathcal{L}(RE(s_i, m_{1:M}^{(n)}))}, m_{1:M}^{(n)} \in Monte_{\pi_\gamma} & j < M \\ e^{-\mathcal{L}(RE(s_i, m_{1:j}))} & j = M \end{cases} \tag{3.9}$$

where $\mathcal{L}$ denotes the cross entropy loss and RE is the relation extraction model. Note that an exponential function is applied to make sure that the sentences group that has lower cross entropy loss has a greater reward value.

3.3.3 Policy Gradient

After agents decide a set of actions based on their policies, the relation extraction model gives the rewards. The objective of an RL algorithm is to maximize the overall expected cumulative rewards by updating policy parameters following a policy gradient strategy [52]. The gradients of our two agents can be computed using Eq. 3.10.

$$\begin{aligned} \bigtriangledown \overline{R}_\theta &= \sum_\tau \sum_\iota R(\iota)\pi_\gamma(\tau,\iota)\pi_\theta(\tau) \bigtriangledown log\pi_\theta(\tau) \\ &= \mathbb{E}_{\tau\sim\pi_\theta(\tau)}[\sum_\iota R(\iota)\pi_\gamma(\tau,\iota) \bigtriangledown log\pi_\theta(\tau)] \\ &\approx \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} R(i,j)\pi_\gamma(b_{i,j}, m_{i,j}) \bigtriangledown log\pi_\theta(a_i, s_i) \\ \bigtriangledown \overline{R}_\gamma &= \sum_\tau \pi_\theta(\tau) \bigtriangledown \sum_\iota R(\iota)\pi_\gamma(\tau,\iota) log\pi_\gamma(\tau,\iota) \\ &= \sum_\tau \pi_\theta(\tau)\mathbb{E}_{\iota\sim\pi_\gamma(\iota)}[R(\iota) \bigtriangledown log\pi_\gamma(\iota)] \\ &\approx \sum_{i=1}^{N}\pi_\theta(a_i, s_i)\frac{1}{M}\sum_{j=1}^{M} R(i,j) \bigtriangledown log\pi_\gamma(b_{i,j}, m_{i,j}) \end{aligned} \tag{3.10}$$

where $\bigtriangledown\overline{R}_\theta$, $\bigtriangledown\overline{R}_\gamma$ denote the derivation of the reward R w.r.t. parameters of main sentence policy π_θ and parameters of supplementary sentence policy π_γ, respectively. Then the parameters of π_θ and π_γ can be updated via Eq. 3.11

$$\begin{aligned} \theta &\leftarrow \theta + \alpha_\theta \bigtriangledown \overline{R}_\theta \\ \gamma &\leftarrow \gamma + \alpha_\gamma \bigtriangledown \overline{R}_\gamma \end{aligned} \tag{3.11}$$

where α_θ and α_γ are the learning rates.

3.4 Experimental Evaluation

3.4.1 Dataset

Table 3.1: An example of using the weaker distant supervision to label WikiText, given the fact from Wikidata Knowledge Base

Fact	{relation: educated at; main entities: Marie Curie, University of Paris; supplementary entities: physics(major), Doctor of Science(degree)}
main sent	In June 1903, **Marie Curie** was awarded her doctorate from the **University of Paris**.
main sent	**Marie Curie** was the first woman to become a professor at the **University of Paris**.
sup sent	In 1893, **Marie Curie** was awarded a degree in **physics** and began work in an industrial laboratory of Professor Gabriel Lippmann.

PubMed The PubMed dataset is created by automatically labeling biomedical literature with Gene Drug Knowledge Database. The labeling process follows this rule: a candidate is retained only if no other co-occurrence of the same entities in an overlapping text span with a smaller number of consecutive sentences [37]. In this dataset, there are 6,987 ternary drug-gene-mutation relation instances and 6,087 binary drug-mutation relation instances. There are 5 categories of relations:"resistance or nonresponse", "sensitivity", "response",

"resistance" and "none". Following previous work [37], we binarize the multi-class relations by replacing the first four relations as "yes". We report the experimental results on the binary relation extraction and on the multi-class relation extraction.

WikiText A complete fact not only appears in consecutive sentences but also in non-consecutive sentences. The strong distant supervision hypothesis used in PubMed only consider consecutive sentences. To consider these two situations at the same time and test whether the proposed model can reduce the impact of noise data in both situations, we also create a new dataset using a weaker distant supervision assumption. We first collect Wikipedia webpages under the "People" category and remove all non-text symbols [56]. Then Wikidata is used as a Knowledge Base to automatically label the relations for these webpages. In Wikidata, each fact consists of two values (main entities), n qualifiers (supplementary entities) with n roles where $n \geq 0$, and one property (relation). The labeling process follows this rule: if the sentence has at least one main entity or two supplementary entities that participate in one specific fact, this sentence possibly indicates the relation of that fact. Specifically, as the example shown in Table 3.1, if the sentence has two main entities, this sentence is labeled as the main sentence of that relation. Others are labeled as supplementary sentences of that relation. Note that using this labeling process, some sentences may be labeled more than one relation, which makes the task more challenging. Compared to distant supervision used in the PubMed dataset, this labeling process is a weaker distant supervision assumption and does not restrict the consecutiveness.

Statistically, there are 2,133 facts, 4,194 main sentences and 13,440 supplementary sentences in the WikiText dataset. The number of different relations is 55, while the number of different roles is 90. We select 20% main sentences and 20% supplementary sentences individually as the test dataset. In this randomized selection, we also make sure that the instance that has sentences in the test dataset also has sentences in the training dataset. This selection

process is applied for 5 times and we report the average accuracy and standard derivation on this dataset.

3.4.2 Baseline Methods

We compare with the following baselines: (a) Graph LSTM-based models, including Graph LSTM-EMBED/FULL/ multitask [37]; (b) Graph state LSTM model(GS LSTM) [50]; (c) LSTM-CNN model, which encodes sentences using LSTM first then extracts features using CNN [29]; (d) Graph Convolutional Networks (GCN) and Attention Guided GCN (AG-GCN) [17, 64]; (e) Multi-head attention-based model model [66]. Besides baselines, the RE model is also tested individually as a variant of the proposed model.

3.4.3 Evaluation Results

Table 3.2: Average test accuracy in five-fold validation on PubMed dataset. Ternary denotes drug-gene-mutation interactions and Binary denotes binary drug-mutation interactions. "-" denotes that the value is not provided herein.

Model	Binary class				Multi-class	
	Ternary		Binary		Ternary	Binary
	Single	Cross	Single	Cross	Cross	Cross
G-LSTM-EMB	76.5	80.6	74.3	76.5	—	—
G-LSTM-FULL	77.9	80.7	75.6	76.7	—	—
G-LSTM MUL	—	82	—	78.5	—	—
LSTM-CNN	79.6	82.9	85.8	88.5	—	—
GCN (K=0)	85.6	85.8	82.8	82.7	75.6	72.3
GS GLSTM	80.3	83.2	83.5	83.6	71.7	71.7
AGGCN	87.1	87	85.2	85.6	79.7	77.4
Multih attn	81.5	87.1	87.4	**89.3**	84.9	80.1
RE model	88.0±0.3	88.3±0.2	89.1±0.2	86.5±0.4	85.1±0.3	80.4±0.2
RE with SentDE	**88.6**±0.1	**89.2**±0.1	**90.1**±0.2	88.7±0.3	**86.7**±0.2	**81.3**±0.2

Evaluation on PubMed

We report average test accuracy in five-fold validation on the PubMed dataset. As shown in Table 3.2, the performance of the proposed model is better than previous SOTA baselines on most tasks[1]. Specifically, the test accuracy of RE model on all ternary relation tasks is higher than baselines, which shows that the RE model is capable of processing multi-entity relation extraction. After training the SentDE model and the RE model iteratively, the impact of noise data on the training process of the RE model is greatly reduced, so that the accuracy of the proposed model is higher than that of the RE model on all tasks. Meanwhile, we notice that the accuracy of most baselines on multi-class tasks is much lower than that on binary-class tasks, e.g., the accuracy of AGGCN is reduced by about 10%. However, our model still maintains a high accuracy even on multi-class tasks, which is 1.8% higher than SOTA.

In the binary entity relation extraction tasks, the performance of our model drops a little. One possible reason is that we apply PCNN to extract the feature of each sentence. In the binary relation data, there are many sentences with only one entity, which does not meet the conditions of PCNN. In the experiment, the second anchor of PCNN on this kind of sentence is set at the beginning of the sentence by default.

Table 3.3: The average test accuracy and standard deviation on WikiText dataset.

Model	Accuracy(%)
LSTM-CNN	37.9 ± 2.5
Multihead Attention	52.2 ± 2.6
RE	59.3 ± 1.7
RE with SentDE(random)	64.7 ± 1.3
RE with SentDE(no indicators)	65.1 ± 0.8
RE with SentDE	**66.4** ± 0.9

[1]Evaluation results of the first four models are reported from the literature

Evaluation on WikiText

Since the baseline model is designed for consecutive sentences, the order of input sentences is set so that the main sentence is the first, followed by all the supplementary sentences in order. This order is also used in our proposed model without the SentDE model.

As shown in Table 3.3, the test accuracy of the RE model is 7.1% higher than the best performance of baselines. This shows that the RE model is more capable of encoding non-consecutive sentences and predicting the relations than previous models. Considering both the results on the WikiText and PubMed dataset, the proposed RE model is a universal model that fits for both non-consecutive and consecutive cross-sentence n-ary relation extraction tasks. Note that the number of relations (classes) is 55, so the 66.4% test accuracy of the proposed model is a fairly great result, which is significantly better than the RE model. This indicates that with the help of SentDE agents, the RE model is more possible to learn the real relation distribution.

Evaluation on SentDE model

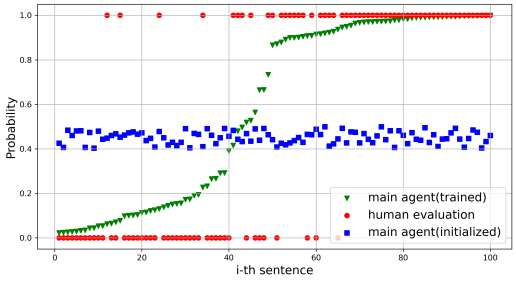

Figure 3.2: The probability distribution of sentences

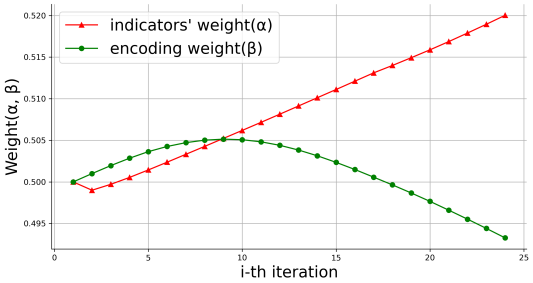

Figure 3.3: The value of weights on each training iteration

We first randomly select 100 main sentences from the test set and ask a graduate student to check whether the relation labeled with distant supervision is correct for these 100 sentences.

The correctly labeled sentences are marked with "1", while others are marker with "0". Given by the main sentence-level agent, the probability that the sentences are correctly labeled is also reported. As shown in Fig.3.2, the probability distribution of the sentences given by the agent has a strong positive correlation with the results of manual inspection. Specifically, most low-probability sentences are marked with "0" by human evaluation, which indicates that these sentences are incorrectly labeled by distant supervision, while most high-probability sentences are correctly labeled. This demonstrates that the well-trained main sentence-level agent can distinguish incorrectly labeled data from correctly labeled data.

To investigate whether the three indicators selected in the supplementary sentence-level agent affect model performance, we tracked the changes of the two weights, α and β, during training and reported them in Fig.3.3. Both weights are initialized to 0.5 and their values change slightly during training. The weight of the three indicators does not approach 0, which demonstrates that the selected three indicators impact the model performance. Table 3.3 shows that the proposed model's test accuracy without these indicators is 1.3% lower than the original proposed model. This also indicates the positive impact of these indicators on model performance. To investigate the impact of the defined transition rule, we replace the transition rule with a random selection process, in which the next state of the supplementary sentence-level agent is randomly chosen from the remaining sentences. Table 3.3 shows that the model accuracy based on this process is 1.7% lower than the original model. This demonstrates that the transition rule based on variety of entities helps the proposed model alleviate the noise data effect.

Chapter 4

Unsupervised Binary Intra-sentence Relation Extraction

4.1 Problem Formulation

The problem of unsupervised binary intra-sentence relation extraction can be defined as follows. Given a text T, the model should learn the clusters of entity pairs (e_i, e_j), based on the relation similarities of their associated sentences. Then, given (e_i, e_j), the model selects the closest centroid from cluster centroid set C and uses the label of that centroid as relation label r_k.

4.2 Clustering-based Approach

4.2.1 Model Overview

The proposed Clustering-based Unsupervised Generative Relation Extraction (CURE) model includes two stages. The first is the relation extractor training stage. We train a relation extraction model, which takes text and (e_i, e_j) as input and outputs vectorized relation representations. The second is the triplets clustering stage. In this stage, the relation extractor model is used to extract relation representations then the relations are clustered.

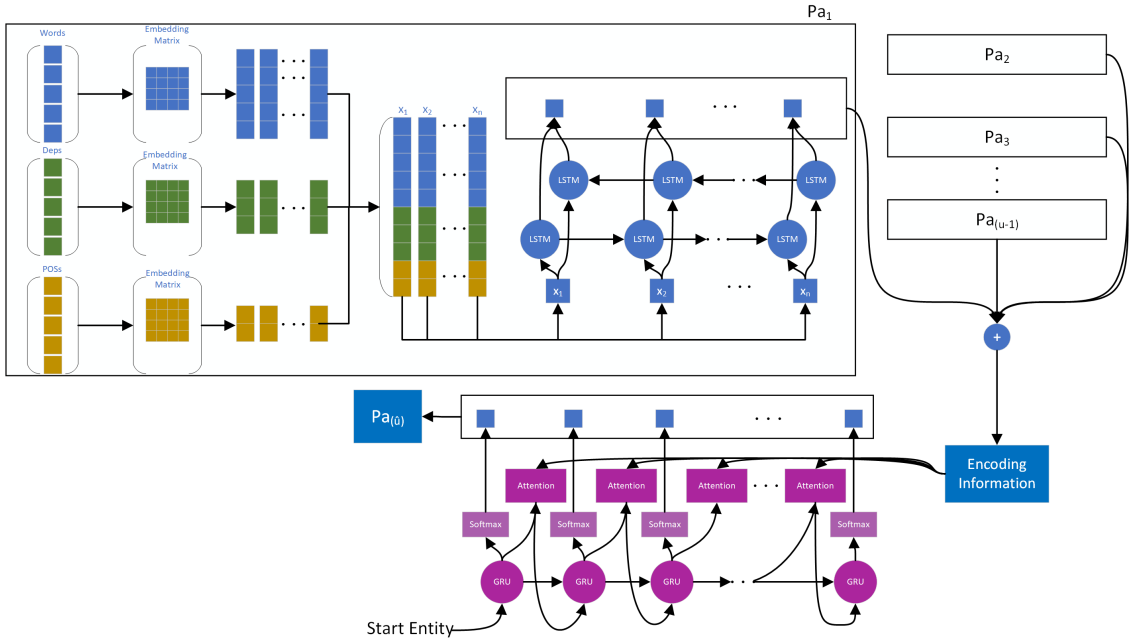

Figure 4.1: The architecture of relation extractor training stage of CURE

After labeling each cluster centroid, for a given (e_i, e_j), the model selects the closest centroid from cluster centroid set C and uses the label of that centroid as r_k.

We begin by introducing the Encoder-Decoder model that is used to train the relation extractor. This proposed model captures the relation information given (e_i, e_j) and text. The model architecture is shown in Figure 4.1. This training model first encodes the semantic shortest paths of one entity pair in various sentences. The encoding information generated by the encoder reflects the relation information of the input (e_i, e_j). The decoder uses the summation of this information to generate the predicted semantic shortest path of that entity pair. More formally, our model optimizes the decoder $(\mathcal{D})$ and encoder $(\mathcal{E})$, s.t.

$$\underset{\mathcal{D}_\theta, \mathcal{E}_\gamma}{\operatorname{argmax}} \mathbb{P}(Pa_u | Pa_1, Pa_2, \cdots, Pa_{u-1}) \tag{4.1}$$

where Pa_i is the i-th semantic shortest path of (e_i, e_j).

The formal definition of semantic shortest path is explained in section 4.2.2. Here, we briefly explain why the task of this stage is to predict $\hat{Pa}_u$ given other semantic shortest paths. Note that it is necessary to build a well-trained encoder that can extract relation information from given semantic shortest paths. However, the training data does not provide correct relations of each entity pair, therefore it is not possible to train the encoder using a supervised approach. Similar to self-supervised learning techniques, the key idea is to find "correct expected result" to let the model fit without labeling the data. In our relation extraction scenario, since all the semantic shortest paths of one entity pair possibly share similar relation information, we treat one of them as the "correct expected result", and the remaining semantic shortest paths are provided as input to the encoder-decoder training model. This "correct expected result" will be generated as output by that model. This proposed semantic shortest path prediction approach provides a mechanism that can train the encoder-decoder model, while making sure this model can converge. The well-trained model indicates that the individual parts, $\mathcal{D}$ and $\mathcal{E}$ are also well-trained, which satisfies our expectation from the relation extractor training stage.

In the triplets clustering stage of CURE, the well-trained encoder is used as the relation extractor. The procedure of using the relation extractor model is shown in Figure 4.2. This procedure first generates encoding information of input entity pairs (e_i, e_j) using the pre-trained relation extractor. Then entity pairs are clustered based on their corresponding encoding information. After labeling each cluster centroid, each entity pair (e_i, e_j) is assigned a relation r_k, which is the cluster label. The details are discussed in Section 4.2.6.

Table 4.1: An example of path search

original sentence	Ronald Reagan served as the 40th president of the United States.
Entity Pair	(Ronald Reagan, the United States)
Dep Path	['nsubj', 'ROOT', 'prep', 'pobj', 'prep', 'pobj']
POS Path	['PROPN', 'VERB', 'ADP', 'NOUN', 'ADP', 'PROPN']
Word Path	['Reagan', 'served', 'as', 'president', 'of', 'States']

4.2.2 Semantic Shortest Paths

Given a dependency tree of one sentence, the semantic shortest path (SSP) of two entities is defined as the shortest path from one entity (node) to the other entity (node) in the dependency tree. Razvan et al. mentioned that the semantic shortest path can capture the relation information of entity pairs [7]. Table 4.1 shows an example in which, given an entity pair and a sentence, the semantic shortest path is the path from the start entity "Ronald Reagan" to the end entity "the United States". Since only words on this path may not be sufficient to capture the relation information, we save the dependency tags D, Part-Of-Speech (POS) tags P and words W to represent this path.

However, since some entities are compound words, which can be divided into different nodes by the dependency parser, we choose the word that has a "subjective", "objective" or "modifier" dependency relation as a representative. For example, we use "Reagan" as the start entity to find the path because the dependency tag of "Reagan" is "nsubj", while the dependency tag of "Ronald" is "compound".

4.2.3 Encoder

For each semantic shortest path of a given entity pair (e_i, e_j), the D, P and W sequences are embedded into vectors with different dimensions. Since words have more variation than POS tags and Dependency tags, we give more embedding dimensions to W. After the embedding

process, the vector representations of W, P and D are concatenated.

We use a Long Short-Term Memory (LSTM) neural network [22] as the basic unit of the encoder model. We use the Bi-directional LSTM (Bi-LSTM) to encode this sequential data. The Bi-LSTM model considers information from both directions of the text and then concatenates the outputs from each LSTM in different directions. The output of the Bi-LSTM model is shown in Equation 4.2:

$$\begin{aligned} \mathbf{h}''_i &= lstm(\mathbf{x}_i, \mathbf{h}_{i-1}) \oplus lstm'(\mathbf{x}_i, \mathbf{h}'_{i-1}) \\ &= (\mathbf{o}_i \odot tanh(\mathbf{c}_i)) \oplus (\mathbf{o}'_i \odot tanh(\mathbf{c}'_i)) \end{aligned} \tag{4.2}$$

where $lstm$ and $lstm'$ are the LSTM and inverse LSTM functions described in Equation **??**. $\mathbf{o}'_i$, $\mathbf{c}'_i$ and $\mathbf{h}'_{i-1}$ denote the parameters of the inverse LSTM.

After all nodes on the shortest path are encoded, the encoder concatenates each hidden state in order. The encoding information is the summation of encoding results of all shortest paths. The formal description is defined in Equation 4.3:

$$\begin{aligned} ei &= \mathbf{h}''_1 \oplus \mathbf{h}''_2 \oplus \cdots \oplus \mathbf{h}''_n \\ EI &= \sum_{j=1}^{u-1} ei_j \end{aligned} \tag{4.3}$$

where n is the length of each shortest path and ei_j is the encoding result of j-th shortest path. EI is the encoding information of one entity pair.

4.2.4 Decoder

In the decoder part, the words on the semantic shortest path must be generated correctly. If the model can generate the correct word sequences (W), this means that the model has also

correctly learned the complex syntax information. Therefore, we do not require the model to generate P and D at the decoder part.

We use a Gated Recurrent Units (GRU) neural network [21] as the basic unit of our proposed decoder. The GRU architecture has similar characteristics to LSTM, with an additional benefit of having fewer parameters. In order to allow the decoder to fully integrate the encoding information when generating W, we introduce the attention mechanism to the decoder. Attention mechanisms can make the model notice only the information related to the current generation task [54]. This enables the model to more efficiently use the input information, which is the encoding information in this case. In general, as shown in Equation 4.4, the attention mechanism is achieved by using attention weights to incorporate encoding information.

$$\begin{aligned} \overline{h}_i &= GRU(\overline{h}_{i-1}, \overline{q}_{i-1}) \\ \overline{q}_{i-1} &= \mathbf{B}\left(\left(\mathbf{A}(\overline{h}_{i-1}) \otimes EI\right) \oplus \overline{q}_{i-2}\right) \\ &= \mathbf{W}_\beta\left(\left((\mathbf{W}_\alpha \otimes \overline{h}_{i-1} + b_\alpha\right) \otimes EI) \oplus \overline{q}_{i-2}\right) \end{aligned} \tag{4.4}$$

where $\overline{h}_i$ is the output of the i-th GRU unit, which is the predicted probability distribution of the word at that position. $\overline{q}_{i-1}$ is the input of the GRU and the weighted information of the previous state and the encoding information. B and A are two different attention matrices that will be learned.

4.2.5 Loss Function

As discussed in the decoder section, each GRU unit outputs a vector that represents the probability distribution for the word at a given position, where the index of each element of

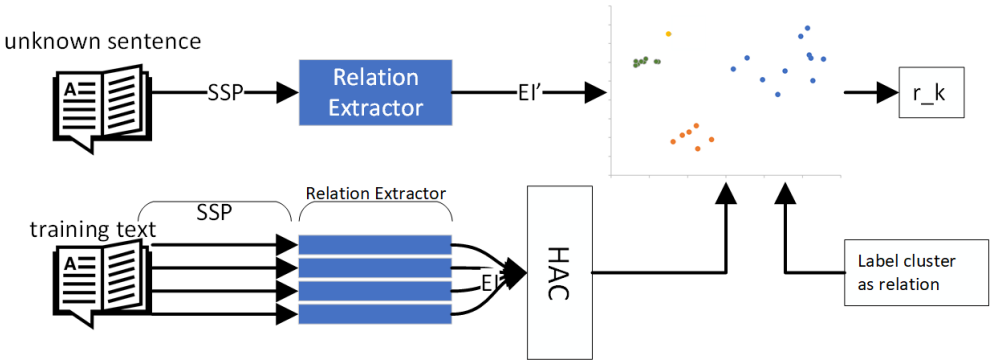

Figure 4.2: The triplets clustering stage of CURE

the vector corresponds to the index of each candidate word.

We design the loss function as the average cross entropy value of each predicted word and correct word. The formal definition of the loss function is in Equation 4.5:

$$\mathcal{J}(\mathcal{D}_\theta, \mathcal{E}_\gamma) = \frac{1}{n}\sum_{i=1}^{n} -log\left(\frac{e^{\overline{h}_{i,k}}}{\sum\limits_{j:j\neq k} e^{\overline{h}_{i,j}}}\right) \tag{4.5}$$

where n is the length of each semantic shortest path. $\overline{h}$ is the output tensor from the decoder. Therefore, $\overline{h}_{i,k}$ indicates the value of the k-th element in the i-th vector.

4.2.6 Triplets Clustering

When training the encoder-decoder model is complete, a relation extractor is obtained, which can extract relation information given semantic shortest paths. The relation extractor can use a vector to represent relation r_k. Therefore, according to the method we introduced in Figure 4.2, we use Hierarchical Agglomerative Clustering (HAC) to cluster similar vectors together using Euclidean distance. The result of the HAC clustering is the same as the clustering result of the entity pairs that share similar relations.

After obtaining these clusters, we extract the W corresponding to the entity pairs in each cluster, thus a candidate relation word set $\mathcal{R}$ is obtained. Based on set R, the relation word of each cluster (i.e., cluster label) can be selected using the Equation 4.6:

$$\begin{aligned} \hat{r_k} &= w \\ &s.t.\ \operatorname*{argmax}_{w} \frac{\phi_e(w) \cdot v}{||\phi_e(w)|| \cdot ||v||} \\ &where\ v = \sum_{r_i \in \mathcal{R}} \mathcal{Z}\left(\sum_{r_j \in \mathcal{R}, j \neq i} \left(1 - \frac{r_i \cdot r_j}{||r_i|| \cdot ||r_j||}\right) C(r_i) \right) r_i \end{aligned} \tag{4.6}$$

where w is the selected relation word, r_i is the vector representation of the i-th word in $\mathcal{R}$ and $C(r_i)$ is the number of occurrences of the i-th word in $\mathcal{R}$. $\phi_e(\cdot)$ is the pre-trained word2vec function. $\mathcal{Z}(\cdot)$ is the min-max normalization function. Our proposed key idea is to first project the words into a high-dimension space using a pre-trained Word2Vec model [33]. Then the vector summation of these words obtains the vector of the relation word.

The direct summation of each word vector will result in loss of important information. However, the more occurrences of a word in $\mathcal{R}$, the weight should be greater in the summation process. For example, suppose "locate" appears ten times and "citizen" appears once in $\mathcal{R}$, which indicates that this cluster is more likely to describe "is located in" than "is citizen of". Thus the model needs to reduce the impact of "citizen". On the other hand, words with more occurrences in $\mathcal{R}$ may also be common words or stop words. Therefore, we add another factor, which measures the cosine similarity between the current word vector and other word vectors in $\mathcal{R}$. If the sum of the cosine similarity is higher, then the word is more similar to other words, so we lower the value of this factor. Here we make an assumption that words that are less similar to other words may be more meaningful. This assumption is based on our observation that many stop words, such as "to" and "from", are close in the vector space.

4.3 Variational Approach

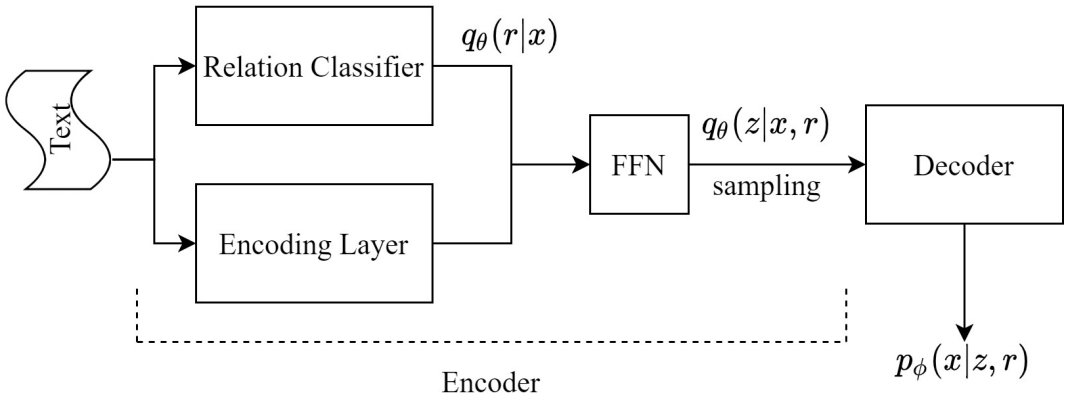

Figure 4.3: The architecture of the proposed model. FFN is a two-layer feed-forward network.

The proposed **V**ariational **A**utoencoder-based **U**nsupervised **R**elation **E**xtraction model (UREVA) uses VAE as the architecture. Specifically, we divide the encoder network into two parts, one is the relation classifier and the other is the encoding layer. The relation classifier calculates the probability that a given input sentence x is classified into relation r, which can be expressed as $q_\theta(r|x)$. We let latent variable z be conditioned on x and r, which can be written as $q_\theta(z|x,r)$. We use the encoding layer to model $q_\theta(z|x,r)$. The decoder network then reconstructs x given samples of z and r, which can be represented as $p_\phi(x|r,z)$. Finally, as shown in Eq. 4.7, our model has the following mathematical property:

$$q_\theta(z,r|x) = q_\theta(r|x)q_\theta(z|x,r) \tag{4.7}$$

4.3.1 VAE-based Objective Function

Variational autoencoder (VAE) is a probabilistic generative model that describes an observation in latent space. The goal of VAE is to train a decoder via $p_\phi(x,z) = p_\phi(z)p_\phi(x|z)$, where $p_\phi(z)$ is a prior distribution of latent variable z and $p_\phi(x|z)$ is the decoder that gen-

erates x given z. In general, since the true posterior distribution $p_\phi(z|x)$ is not tractable, an encoder is used to approximate it, which can be expressed as $q_\theta(z|x)$. The objective of VAE is to minimize the KL divergence of $q_\theta(z|x)$ and $p_\phi(z|x)$ such that the two distributions are similar to each other [55]. Expanding this objective function can give following equation:

$$log(p_\phi(x)) = KL(q_\theta(z|x)||p_\phi(z|x)) + \mathcal{L}(\theta, \phi; x) \tag{4.8}$$

Given this equation, minimizing the KL divergence is equivalent to maximizing $\mathcal{L}(\theta, \phi; x)$. Then the objective function can be derived as follows:

$$\begin{aligned} & log(p_\phi(x)) \geq \mathcal{L}(\theta, \phi; x) \\ & = \mathbb{E}_{q_\theta(z|x)}[log(p_\phi(x|z)) - log(q_\theta(z|x))] \\ & = \mathbb{E}_{q_\theta(z|x)}[log(p_\phi(x|z))] - KL(q_\theta(z|x)||p_\phi(z)) \end{aligned} \tag{4.9}$$

The objective function of our model can be derived from the original VAE objective function by substituting r as follows:

$$log(p_\phi(x, r)) \geq \mathbb{E}_{q_\theta(z,r|x)}[log(p_\phi(x|r, z))] - KL(q_\theta(z, r|x)||p_\phi(z)) \tag{4.10}$$

The goal of our proposed model is to optimize the lower bound of $logp_\phi(x, r)$, which is similar to the goal of the original VAE. To optimize this lower bound, we treat the term on the right hand side of Eq.4.10 as the objective function. We then substitute Eq. 4.7 into the objective function. This enables us to rewrite the objective function as shown in Eq. 4.11:

$$\mathcal{L} = \mathbb{E}_{q_\theta(r|x)}[\mathbb{E}_{q_\theta(z|x,r)}[logp_\phi(x|r, z)] + KL(q_\theta(z|x, r)||p_\phi(z))] + \sum_r \sum_z q_\theta(z|r, x)\mathcal{H}(q_\theta(r|x)) \tag{4.11}$$

where $\mathcal{H}(\cdot)$ represents the entropy function.

4.3.2 Approximation of Objective Function

In this section, we discuss that it is not possible to compute the exact objective function and present two methods to approximate the objective function.

Decoding Approximation

As seen in Eq. 4.11, the goal of the decoder is to compute the probability $logp_\phi(x|r,z)$ of a sentence given a relation and the latent variable. A key challenge is that computing this probability makes it harder to train the model. This is because generating sentences using sampled data is very unstable. To overcome this challenge, we approximate the probability $logp_\phi(x|r,z)$ as the probability of an entity pair $logp_\phi(e_{head}, e_{tail}|r,z)$ given a relation and the latent variable. This entity pair probability can be computed via Eq. 4.12 as follows:

$$p_\phi(e_{head}, e_{tail}|r,z) = \frac{\phi(e_{head}, e_{tail}|r,z)}{\sum\limits_{i,j \in E; i \neq j} \phi(e_i, e_j|r,z)} \tag{4.12}$$

where $\phi(\cdot)$ is the score function modeled by the decoder and E is the set of all entities.

A key challenge is the calculation of the denominator of Eq. 4.12 is computationally intensive. Following [48], we apply negative sampling to approximate $p_\phi(e_{head}, e_{tail}|r,z)$. Specifically, we randomly sample some entities as the input of the decoder. Then the decoder should give high scores to the correct entity pairs and low scores to the randomly formed entity pairs.

Formally, as shown in Eq. 4.13, $p_\phi(e_{head}, e_{tail}|r, z)$ is equivalent to:

$$\begin{aligned} p_\phi(e_{head}, e_{tail}|r, z) \propto & -log(\sigma(\phi(e_{head}, e_{tail}|r, z))) \\ & - \sum_k \mathbb{E}_{e_i \sim p_E}[log(\sigma(-\phi(e_{head}, e_i|r, z)))] \\ & - \sum_k \mathbb{E}_{e_i \sim p_E}[log(\sigma(-\phi(e_j, e_{tail}|r, z)))] \end{aligned} \tag{4.13}$$

where σ is the sigmoid function and k is the sampling times. e_i and e_j are sampled entities based on empirical entity distribution p_E.

Encoding Approximation

To enable VAE to pass the gradient during random sampling, a common method is to use reparameterization to simulate $q_\theta(z|r, x)$ [24]. This method lets the encoder generate the mean μ and variance σ vectors. The result of sampling can then be defined as $\hat{z} = \mu + \epsilon\sigma$, where random variable ϵ following $\mathcal{N} \sim (0, 1)$.

By utilizing this trick, we can calculate the KL divergence term in Eq. 4.11. However, $q_\theta(z|r, x)\mathcal{H}(q_\theta(r|x))$ is not tractable since the probability $q_\theta(z|r, x)$ is unknown. In practice, we replace $q_\theta(z|r, x)$ with a small constant c. This approach is equivalent to changing the different weights of the entropy of different r's into a constant.

4.3.3 Encoder Architecture

According to previous work, entity types are the most significant features that represent the relation information of sentences [53]. We follow this work and apply two different feed-forward networks to model the relation classifier and encoding layer.

Specifically, given sentence x, we extract the entity types of the head and tail entities in x, denoted as t_h and t_t. We use one-hot vector to represent the combination of two entity types $t_h \oplus t_t$. That is, if there are n different entity types in the dataset, then the length of the one-hot vector is n^2. Then relation classifier and encoding layer are modeled via the following equation:

$$\begin{aligned} r &= W_r^T(t_h \oplus t_t) + b_r \\ \hat{x} &= W_e^T(t_h \oplus t_t) + b_e \\ z &= W_{en}^T(\hat{x} \oplus r) + b_{en} \\ \hat{z} &= RT(z, \epsilon) \end{aligned} \tag{4.14}$$

$\hat{z}$ is the sampled data and $\hat{x} \oplus r$ is the concatenation of $\hat{x}$ and r. Note that z is a vector of μ and σ and $RT(\cdot)$ represents reparameterization.

4.3.4 Decoder Architecture

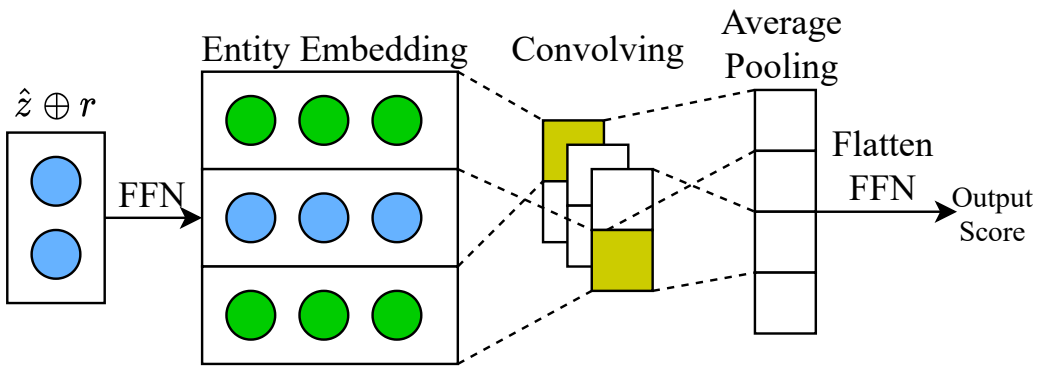

Figure 4.4: The decoder architecture

RESCAL is a three-way tensor decomposition method, which needs to build a three-dimensional matrix to represent the relation embedding. Unlike the previous methods that use RESCAL [35] and selective preference as the decoder, we use a simple CNN to reconstruct triplets. The reason for using CNN is that our decoder needs to sample from the $q_\theta(z|r, x)$ distribution to

establish a connection with the encoder. The reason is that the dimension of the sampling result of z is much smaller than the dimension of a three-dimensional matrix, so forcibly mapping the sampling result to this matrix will cause the matrix to be sparse.

In order to apply CNN, we first concatenate sampled $\hat{z}$ and r and map the result to $h_r \in \mathbb{R}^{n_d}$. This process models the $p_\phi(x|r,z)$, since the decoder should take both r and z into account when reconstructing the triplets according to that term. As shown in Fig 4.4, then we concatenate e_{head}, h_r, e_{tail} as $c_{in} = e_{head} \oplus h_r \oplus e_{tail} \in \mathbb{R}^{3\times n_d}$, where e_{head} and e_{tail} are the embeddings of head entity and tail entity. CNN uses n_f filters, each of kernel size $\mathbb{R}^{2\times n_d}$, to extract features from c_{in}. Then a average pooling is applied to process the results. The flattened average pooling output $c_{out} \in \mathbb{R}^{n_f}$ is mapped to a real number via a linear layer.

4.3.5 Key Insights

Previous work uses encoder as a relation classifier, which outputs a relation classification, a probability distribution r [30]. In their work, the decoder reconstructs triplets based on r and the loss function can be defined via Eq. 4.15

$$\mathcal{L} = \mathbb{E}_{q_\theta(r|x)}[logp_\phi(e_i|e_{-i},r)] + \mathcal{H}(q_\theta(r|x)) \tag{4.15}$$

where e_i is the predicted entity and e_{-i} is the given entity. The model reconstructs the triplet by predicting the missing entity. If we compare Eq. 4.15 with the loss of VAE, one can find that r in Eq. 4.15 is the latent variable z in VAE. In addition, the entropy of r is actually equivalent to the KL divergence of $p_\phi(r|x)$ with uniform distribution. Therefore, unlike the general VAE methods that predefine z as a normal distribution, the loss function defined here pushes $p_\phi(r|x)$ to the uniform prior distribution. In most cases, the encoder does not really classify relation given the input sentence x, because the loss function only requires

reducing the reconstruction loss while maintaining the uniform distribution of $p_\phi(r|x)$. In addition, in their work, when reconstructing the triplet, decoder did not sample data from $p_\phi(r|x)$. Therefore, it is possible that the decoder can successfully reconstruct the entire triplets with uniform $p_\phi(r|x)$. Moreover, a large part of the entities only appeared once in the dataset, which simplifies training the decoder. This explains the question raised in the work of Simon et al [48] that why the model is unstable and likely to classify all input sentences into the same relation.

In order to tackle this problem, 1) $q_\theta(r|x)$ cannot be used to mimic the prior distribution. 2)decoder has to be connected with encoder via the sampling process. Then the main difference between our proposed model and previous approaches is that we regard r as an intermediate variable instead of latent variable, which is conditioned on the real input x. We guarantee from two aspects that the proposed model will really learn relation classification. 1) According to Eq. 4.7, if the encoder can map the input x to latent space z, it must learn $p_\theta(x|r)$, because this is an essential step for the mapping process. 2) According to $q_\phi(x|z,r)$, our decoder reconstructs triplets based on sampled z and r, while the decoder of the previous model does not receive any information from the encoder. This ensures that the decoder utilizes the information from sampled data. Therefore, the relation classifier can receive the gradients to update.

4.4 Experimental Evaluation

4.4.1 Dataset

CURE model

Datasets We use a New York Times (NYT) dataset [43] and the United Nations Parallel Corpus (UNPC) dataset [67] to train and test our model and other unsupervised relation extraction baseline methods.

NYT dataset. In the NYT dataset, following the preprocessing in Rel-LDA, 500K and 5K sentences were selected as the training and testing sets, respectively. Each sentence contains at least one entity pair. Note that only entity pairs that appear in at least two sentences were included in the training set, so the number of entity pairs in training set is 60K. Furthermore, all entity pairs in the testing set have been matched to Freebase [5]. That is, for a given entity pair (e_i, e_j), we have a relation r_k from Freebase.

UNPC dataset. The UNPC dataset is a multilingual corpus that has been manually curated. In this dataset, 3.2M sentences were randomly selected from the aligned text of the English-French corpus and used as the training set. The number of entity pairs in training set is 200k. We selected 2.6k sentences to use as the testing set. Each sentence also contains at least one entity pair. The number of unique entity pairs is 1.5k in the testing set (previous work used a testing set with 1k unique entity pairs [60]). Similarly, all entity pairs in the testing set have been matched to YAGO [42].

While previous state-of-the-art methods for this problem used only the NYT dataset for evaluation, we chose to additionally use this corpus for further evaluation for two reasons: (1) The scale of this dataset is far greater than that of NYT dataset, so the model is more

likely to learn methods for extracting relation patterns. (2) To ensure model robustness and ensure that a model that achieves excellent results on NYT is not over fitting to the dataset.

UREVA model

Data: Following previous work [53], we use NYT-FB dataset to evaluate our model. NYT-FB dataset is obtained by using Freebase to label the corpus of the New York Times. That is, if the entity pair that appears in a sentence also appears in Freebase [5], then this sentence is automatically labeled as the relation stored by Freebase. After filtering out some sentences using syntactic patterns, there are 2 million sentences in the dataset, of which 41,000 are labeled with meaningful relations[1]. Of the 41,000 tagged sentences, 20% are used as validation set, and 80% are used as test set.

As argued in previous work [53], NYT-FB dataset may not be a perfect dataset to evaluate models since the relation is a long tail distribution in this dataset. This allows a model to achieve high performance by predicting each sentence into a unique relation, which is unexpected. Therefore, we also conduct experiments on the other dataset, SemEval dataset [19]. We use SemEval 2010 Task 8, which is Relation Extraction task between pairs of nominals. There are 8,000 sentences in the training set, the entities of each sentence are manually labeled and the relations of these entities are also manually annotated. This dataset has a total of 10 relations, including "Others" that represents no normal relation detected in the sentence. As the previous work has the same preprocessing process for NYT-FB dataset, we use 20% of these sentences as the test set.

[1]The detailed preprocessing steps are described in the previous work [30].

4.4.2 Evaluation Metrics and Baselines

CURE model

Baseline Models: We compare CURE to three state-of-the-art unsupervised relation extraction models. 1 (Rel-LDA): the topic distribution in LDA is replaced with triplets distribution, and similar relations are clustered using Expectation Maximization [62]. 2 (VAE): the variational autoencoder first predicts semantic relation given entity pairs then reconstructs entities based on the prediction. The model is jointly trained to minimize error in entity recovering [30].3 (Open-RE): corresponding sentences of entity pairs are used as features and then the features are vectorized to evaluate relation similarity [10].

UREVA model

Evaluation Metrics: B-cube (B^3) [1], V-measure [44] and Adjusted Rand Index (ARI) [23] are used as evaluation metrics. B^3 is the harmonic mean score of recall and precision of clustering result. Similarly, V-measure is the harmonic mean between homogeneity and completeness, while ARI is anther general way to evaluate clustering performance.

Baselines: In order to compare the proposed approach with other state-of-the-art methods, we use following four models as the baselines. 1: Rel-LDA [62] is designed for relation discovery task based on LDA model [4]. The topic distribution is replaced by relation distribution, modeled by Dirichlet prior distribution. 2: March [30] is a VAE-like model, which encodes relation classification and train a decoder to reconstruct the triplet of given sentence. Note that in later work, this model was boosted by adding regularization terms. Therefore, we compare our method with this improved model, March($L_s + L_d$). 3: Simon [48] proposed using PCNN as the encoder and boosted previous work by adding regularisation

loss terms. 4: EType+ is a straightforward model that takes combination of entity types in each sentence as the input [53]. These combinations are then mapped to the relation classification results through feed-forward networks. Note that for 3 and 4 baseline models, we choose the re-implemented version, which is publicly available [53].

4.4.3 Evaluation on Clustering-based Approach

Results on NYT

Table 4.2 shows the performance of each model on assigning relations to entity pairs, which involves relation extraction followed by clustering. We compare the models on selected relations, which appear most frequently in the testing dataset. We report recall, precision and F1 scores for each method in Table 4.2. Since the original Rel-LDA and VAE methods did not investigate automatic cluster labeling, we compare against a variant of these methods, where we use the most frequent trigger word in each cluster as the label. Trigger words are defined by the non-stop words on semantic shortest paths. A cluster (and each entity pair in that cluster) is labeled by the relation (in Freebase) that is similar to the most frequent trigger word in that cluster. For a given entity pair with two or more relations in Freebase, the predicted relation of this entity pair is considered accurate as long as it matches one of the corresponding relations in Freebase. Notably, CURE achieves the highest accuracy assigning relations to entity pairs as shown in Table 4.2. We also report the F-1 gain in Figure 4.5. Overall, CURE outperforms all other methods with a gain in F-1 score of average 10.47%.

While both our method and VAE involve an encoding and decoding process, there is a key difference between the two methods. CURE considers the correlation of sentences that have the same entity pair, while VAE directly projects the relation information into a high-dimensional space, and reconstructs triplets according to the projection results to train the

encoder. The results show that the CURE relation information extractor is more accurate than VAE. We conjecture that CURE's achieved accuracy improvement is because doing sentence correlation into the model is equivalent to guiding the converge direction when training the encoder. We note that it can be difficult to clearly distinguish some relations in a sentence. For example, the two clusters for "placeBirth" and "placeLived" partially overlap, so the F-1 score of each model on these two relations is relatively low. In future work, we plan to further investigate and address this finding.

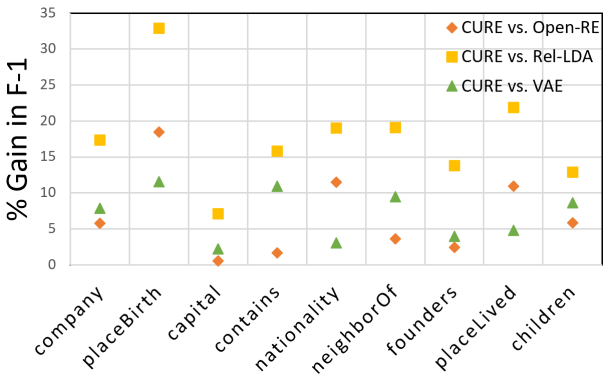

Figure 4.5: % F-1 gain of CURE over baselines on NYT

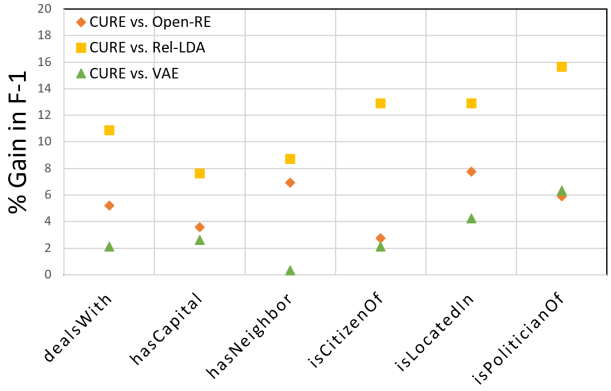

Figure 4.6: % F-1 gain of CURE over baselines on UNPC

Results on UNPC

We use the same experimental settings and parameters we used on the NYT data set. Similarly, Table 4.3 reports recall, precision and F1 scores and shows that our model achieved the best performance in most relations. Note that the genre of UNPC (political meetings records) is different from that of NYT. Therefore, the relations in UNPC are mainly based on national relations and geographical location. Although, overall, CURE outperforms all the baselines, we note that it did not perform well on some relations. In these cases, we notice that CURE performs more detailed clustering than needed. For example, given the relation "isPoliticianOf", CURE divides entity pairs in this category into finer grain subsets, such

Table 4.2: Experimental results on NYT

Relation	System	Rec.	Prec.	F1
company	**CURE**	**48.2**	**60.4**	**53.6**
	Open-RE	46.8	54.9	50.5
	Rel-LDA	39.4	50.7	44.3
	VAE	47.3	51.6	49.4
placeBirth	**CURE**	**47.5**	**38.2**	**42.3**
	Open-RE	38.4	31.3	34.5
	Rel-LDA	31.7	25.7	28.4
	VAE	43.2	32.9	37.4
capital	**CURE**	54.2	65.5	**59.3**
	Open-RE	53.2	**66.1**	59.0
	Rel-LDA	48.4	63.9	55.1
	VAE	**56.3**	59.8	58.0
contains	**CURE**	**56.7**	53.4	**55.0**
	Open-RE	51.6	**56.9**	54.1
	Rel-LDA	43.3	49.8	46.3
	VAE	49.1	49.0	49.0
nationality	**CURE**	39.8	**75.4**	**52.1**
	Open-RE	36.4	62.8	46.1
	Rel-LDA	31.3	64.6	42.2
	VAE	**41.3**	65.1	50.5
neighborOf	**CURE**	**43.9**	**45.1**	**44.5**
	Open-RE	42.5	43.4	42.9
	Rel-LDA	33.8	38.6	36.0
	VAE	37.1	44.0	40.3
founders	**CURE**	**46.4**	45.3	**45.8**
	Open-RE	45.1	44.4	44.7
	Rel-LDA	35.9	43.9	39.5
	VAE	42.6	**45.5**	44.0
placeLived	**CURE**	**38.7**	**33.1**	**35.7**
	Open-RE	37.4	27.6	31.8
	Rel-LDA	32.4	24.5	27.9
	VAE	35.3	32.9	34.0
children	**CURE**	52.8	**47.0**	**49.7**
	Open-RE	48.0	45.7	46.8
	Rel-LDA	44.3	42.3	43.3
	VAE	**53.1**	39.7	45.4

Table 4.3: Experimental results on UNPC

Relation	Models	Rec.	Prec.	F1
dealsWith	**CURE**	67.3	**56.6**	**61.5**
	Open-RE	62.7	54.4	58.3
	Rel-LDA	60.3	50.3	54.8
	VAE	**67.5**	54.3	60.2
hasCapital	**CURE**	**62.9**	**60.2**	**61.5**
	Open-RE	60.5	58.1	59.3
	Rel-LDA	56.7	56.5	56.8
	VAE	61.6	58.3	59.9
hasNeighbor	**CURE**	**68.5**	**56.7**	**62.0**
	Open-RE	62.3	53.8	57.7
	Rel-LDA	61.4	52.6	56.6
	VAE	67.3	54.6	61.8
isCitizenOf	**CURE**	**57.6**	40.1	**47.3**
	Open-RE	55.2	39.5	46.0
	Rel-LDA	52.5	36.9	41.2
	VAE	53.1	**41.0**	46.3
isLocatedIn	**CURE**	**71.9**	**46.7**	**56.6**
	Open-RE	68.7	42.1	52.2
	Rel-LDA	66.0	39.4	49.3
	VAE	68.3	44.9	54.2
isPoliticianOf	**CURE**	**47.5**	**41.1**	**44.1**
	Open-RE	44.7	38.8	41.5
	Rel-LDA	39.2	35.7	37.2
	VAE	45.2	38.0	41.3

as "president" or "ambassador". We also report the F-1 gain in Figure 4.6. Overall, CURE outperforms the other methods with an average F-1 score gain of 6.59%. Experiments on UNPC show that CURE outperforms SOTA approaches and generalizes better to different domains and data sizes.

Clustering Performance

We evaluate clustering performance of each model using rand index. We implement the evaluation as follows: 1) We pair n entity pairs in the testing set together. Therefore, we

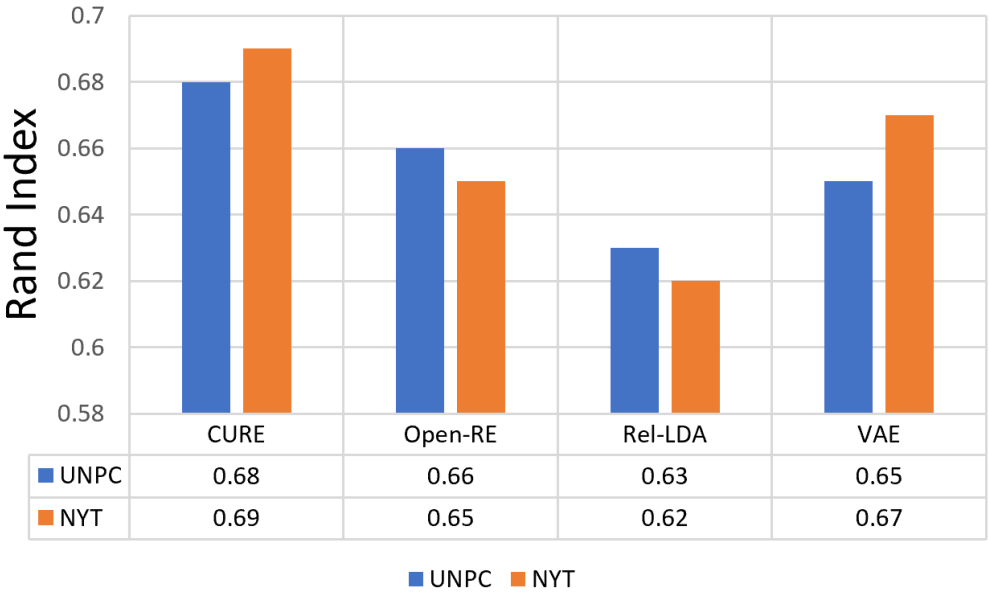

Figure 4.7: Rand Index score of CURE and baselines

obtain $\binom{n}{2}$ pairs of entity pairs. 2) We partition the testing set into m subsets using Freebase or YAGO, and into k subsets using CURE and the baseline methods. Following the definition of rand index, we then compare the m and k subsets to measure the similarity of the results of the two partitioning methods.

The rand index evaluation result is shown in Figure 4.7. Overall, CURE outperforms state-of-the-art methods on both datasets. CURE performs slightly better on NYT than on UNPC. One possible reason is that most sentences of the UNPC dataset do not directly explain the relation between two entities, so some entity pairs are assigned to more general relations, such as "contains".

Label Words Selection Evaluation

In this section, we compare the results of two approaches for **selecting relation words**: (1) based on word vector similarity (denoted as **WVS** and used by CURE), and (2) based on common words (denoted as **CW** and used by previous work [18]). Other approaches that

Table 4.4: Clustering Label Comparison between selecting relation words based on word vector similarity (WVS) and selecting relation words based on common words (CW)

	Label Words	Relation
WVS	**metropolis government city**	capital
CW	city states help	
WVS	**live stay york**	placeLived
CW	york live play	
WVS	**born rise country**	placeBirth
CW	country city live	
WVS	**near neighbor close**	neighborOf
CW	include like york	
WVS	**business executive group**	company
CW	group expert executive	
WVS	**locate include states**	contains
CW	states country city	

rely on experts to manually specify relation words based on extracted trigger words are not included in this comparison. We implement this evaluation as follows: (1) For each relation r_f in Freebase, we count the number of entity pairs with the relation r_f in each cluster. (2) We select the cluster that contains the most entity pairs with the relation r_f. (3) WVS and CW are used to generate the label of the selected cluster. (4) We compare the top three generated cluster labels with the relation r_f as shown in Table 4.4.

The relation words selected by WVS can capture the relations better than CW. For example, for the relation "contains", WVS finds words that describe the relation between two geographic locations, such as "locate" and "include". However, CW can only find that "contains" is related to each geographical division, such as "state" and "country". Moreover, the candidate word lists generated by WVS and CW have different orders. For example, for the relation "company", CW regards "group" as the best word to describe the relation and puts "executive" in the last place. This arrangement is obviously not consistent with facts, because "company" in Freebase mainly emphasizes the relation between the company's leader or owner and the company. WVS arranges its candidate words list differently and

more accurately, putting "business" in the first place and "executive" in the second place. Finally, both label clustering methods are affected by the noise in the text. For example, for the relation "placeLived", both CW and WVS mistakenly included "york" as a candidate relation word because "New York Times" appeared many times in the NYT dataset.

4.4.4 Evaluation on Variational Approach

Table 4.5: The evaluation results of UREVA and baseline methods on NYT-FB and SemEval dataset. Note that $r = i$ indicates that there are i clusters in each model. r on the left of the table corresponds to the setting of NYT-FB, and r on the right of the table corresponds to the setting of SemEval. – denotes that the result is not provided herein.

		NYT-FB			SemEval			
	Model	B^3	V-measure	ARI	B^3	V-measure	ARI	
r=10	RelLDA	29.1	37.4	24.2	–	–	–	r=10
	March($L_s + L_d$)	37.5	38.7	26.1	–	–	–	
	Simon	32.6	30.5	23.8	22.3	11.2	9.7	
	EType+	41.9	40.6	30.7	–	–	–	
	UREVA	**43.1**	**42.0**	**31.6**	**24.5**	**13.8**	**11.7**	
r=16	March($L_s + L_d$)	36.9	37.4	28.1	–	–	–	r=5
	Simon	–	–	–	21.6	11.5	10.6	
	EType+	41.5	41.3	30.5	–	–	–	
	UREVA	**43.4**	**41.6**	**31.5**	**25.1**	**14.4**	**12.1**	
r=100	RelLDA	29.6	37.7	25.1	–	–	–	r=15
	Simon	–	–	–	19.8	10.9	9.8	
	March($L_s + L_d$)	35.8	35.4	27.3	–	–	–	
	UREVA	**41.9**	**43.2**	**29.0**	**23.5**	**13.8**	**10.3**	

Evaluation Metrics Results

Performance on NYT-FB: Table 4.5 shows the average results across three-runs of each model. The performance of UREVA on NYT-FB dataset is better than that of state-of-the-art models. Note that as presented in previous work, the performance of Simon model

that replaces PCNN with entity type is still not as good as EType+ [53]. Therefore, we focus more on the performance of UREVA and EType+ at $r = 10$ and $r = 16$. In general, UREVA's performance is better than EType+ in both cases. In addition, as r increases, UREVA's performance has a slight improvement. This is expected because the more relation clusters predicted by the model, the more likely it is to fit the true relation distribution. In comparison, the performance of EType+ has not steadily improved as r increases. For example, the values of V-measure and B^3 drop marginally.

Performance on SemEval: We also report the performance of UREVA and Simon models on the SemEval dataset. Note that in experiments, we found that most of the entities labeled in the SemEval dataset are not named entities. That is, all models based on predifned features, such as entity types and dependency path, cannot be evaluated on this dataset. Recall that the encoder of our model is also based on entity types, therefore, in order to compare with Simon's model on this dataset, we replaced the encoder architecture with PCNN, which is the same architecture used in the Simon model. In addition, since there are only 10 different relations in the SemEval dataset, we set the number of clusters of the models to 5, 10, and 15.

Both Simon and UREVA's evaluation values on this dataset have dropped significantly, which is reasonable however. The possible reason is that the number of relations in SemEval is far less than the number of relations in NYT-FB. The reduction in the number of relations will cause any wrong classification to have a great impact on the evaluation value. For example, the V-measure score of random classification is about 15 in NYT-FB dataset. For comparison, the V-measure score of random classification on SemEval dataset is only 0.4. Compared to the V-measure score of random classification on SemEval dataset, the gain of UREVA and Simon's V-measure score are 33.5, 27, respectively. The gain of UREVA and Simon's V-measure score are 1.8, 1.03 on the NYT-FB dataset, respectively. Therefore,

the classification accuracy of the two models did not decrease significantly. In addition, compared with Simon, UREVA can still maintain a relatively high classification accuracy on SemEval dataset.

Analysis on Classification Accuracy

As indicated in Section 4.4.4, we found that even if the model keeps predicting the input sentences into the same relation, the B^3 score of the model still remains around 22 on NYT-FB dataset. Similarly, if we randomly classify the input sentences, the V-measure score of the classification result also exceeds 15. We note that this is a key observation. Based on this observation, it is not clear whether the three evaluation metrics used in previous works (B^3, V-measure, ARI) present a true measure of model performance [48, 53]. Therefore, in order to answer this question, we next analyze each relation clustering predicted by the model to ensure that the relation classification is indeed accurate.

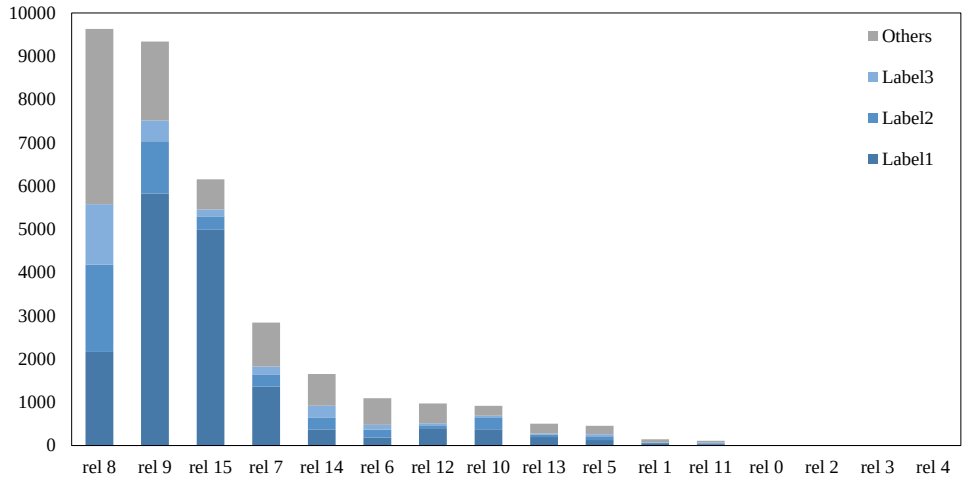

Figure 4.8: Predicted relation groups. rel_i is i-th predicted relation group. Label1 is the real relation that appears the top most frequently in a relation group. Label2 and Label3 represent the real relation that appears second and third most frequently in a relation group, respectively. The ordinate represents the number of sentences classified into each relation group.

As shown in Figure 4.9, in the NYT-FB test set, the relation distribution is similar to a long-tailed distribution. A small number of relations have a high frequency and most of the relations appear with very low frequency in the dataset. For example, the first three relations with the most occurrences account for nearly 50% of the total relations. The result of the relation distribution predicted by our model is similar to this fact. As shown in Figure 4.8, we list the relation distribution output by the model on the test set and the number of different relations output by the model is set to 16. One can see that the relation distribution predicted by the model has a similar long-tailed distribution shape to the actual relation distribution. We also list the top three real relations in each predicted relation group, and label them as label1, label2 and label3. For example, suppose 50 sentences in the test set are predicted to be relation $r_p^{(0)}$. Among these 50 sentences, 16 sentences are labeled as actual relation $r_r^{(1)}$, 15 sentences are labeled as $r_r^{(2)}$, 14 sentences are labeled as $r_r^{(3)}$, and 5 sentences are labeled as $r_r^{(4)}$. Then $r_r^{(1)}$, $r_r^{(2)}$ and $r_r^{(3)}$ are label1, label2 and label3, respectively. If the first three labels account for a high proportion of each predicted relation group, it means that the model does not randomly classify sentences into different relations. The reason is that random classification will give a uniform distribution to each predicted relation group.

Table 4.6: An example of semantic meanings of top-frequency real relations of each predicted relation group.

Relation 5	Relation 9
founders	placeLived
directedBy	nationality
authorEeditor	containedby
writtenBy	placeOfBirth
child	company
owner's	placeOfDeath
containedby	placeOfPublication
majorShareholders	placeOfBurial
worksWritten	country

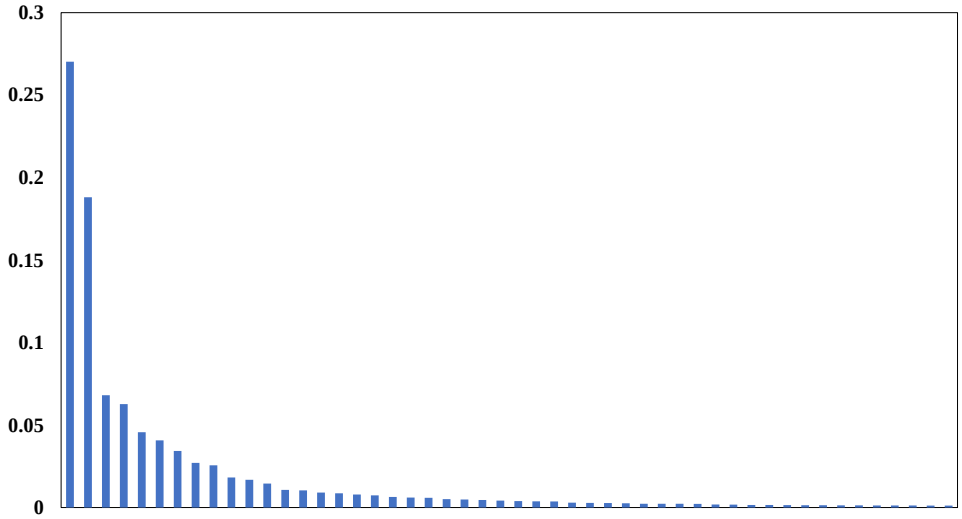

Figure 4.9: The real relation groups. The ordinate represents the percentage of the number of sentences in each relation group to the number of sentences in the dataset. The x-axis is the relations sorted according to the number of sentences contained. For ease of observation, the x-axis label is omitted

Next, we provide a qualitative analysis, which shows that the different relation classes predicted by UREVA have different meanings. As shown in Table 4.6, we randomly select two of the predicted relation groups, relation 5 and relation 9, and find the top 9 real relations that appear most frequently in each of these two relation groups. By analyzing the semantics of these real relations, it is not difficult to find that relation 5 mainly describes the subordination relationship between people or objects. Conversely, relation 9 describes the relationship between people or objects and a geographic location. This shows that different relation groups predicted by UREVA represent different relation information semantically.

Chapter 5

Unsupervised N-ary Cross-sentence Relation Extraction

5.1 Problem Formulation

Unsupervised n-ary cross-sentence relation extraction is a special case of unsupervised relation extraction task. We formulate this problem as follows: given an input text $S = \{s_1, s_2, \cdots, s_n\}$, where s_i is the sentence of this text, the unsupervised n-ary cross-sentence relation extraction model determines which sentences can form a sentence group, which contains an n-ary relation. Then the model should cluster the sentence groups that have the same relation together. In other words, there are two technical problems that need to be solved.

1. Train an unsupervised relation extraction model that can cluster sentence groups with the same relations.

2. Train a model that learns how to determine whether the input sentence group is valid.

5.2 Proposed Approach

In order to tackle with theses two problems, we propose **S**election-**G**uided **V**ariational **A**utoencoder-based **U**nsupervised **R**elation **E**xtraction (SG-UREVA) model. The proposed model consists of two components, the selector and the Variational Autoencoder (VAE). The selector model learns to select high-quality sentence group from the input and remove the sentence group that does not describe a relation. The VAE model is similar to UREVA model, which trains a relation classifier as the encoder that clusters the sentence groups that have the same relation.

5.2.1 Loss Function of VAE

We first expand the loss function defined in UREVA model. Here we assume that the latent variable z is conditioned on the joint distribution of input sentence group x, relation classification r, and the selector results s. That is, both s and r are the intermediate variable of the VAE model. Therefore, the loss function of the VAE model can be define in Eq. 5.1:

$$log(p_\phi(x,r,s)) \geq \mathbb{E}_{q_\theta(z,r,s|x)}[log(p_\phi(x|r,s,z))] - KL(q_\theta(z,r,s|x)||p_\phi(z)) \tag{5.1}$$

where q_θ is the encoder and p_ϕ is the decoder. KL represents the Kullback–Leibler divergence.

As we did in UREVA model, we can rewrite this loss function to split $q_\theta(z,r,s|x)$. The new loss function is shown in Eq. 5.2

$$\begin{aligned}\mathcal{L} = q_\theta(s|x)\mathbb{E}_{q_\theta(r|s,x)}[\mathbb{E}_{q_\theta(z|x,r,s)}[logp_\phi(x|r,s,z)] + KL(q_\theta(z|x,r,s)||p_\phi(z))] - \\ q_\theta(s|x)log(q(s|x))q(z|r,s,x) + q_\theta(s|x)\sum_r\sum_z q_\theta(z|r,x,s)\mathcal{H}(q_\theta(r|s,x))\end{aligned} \tag{5.2}$$

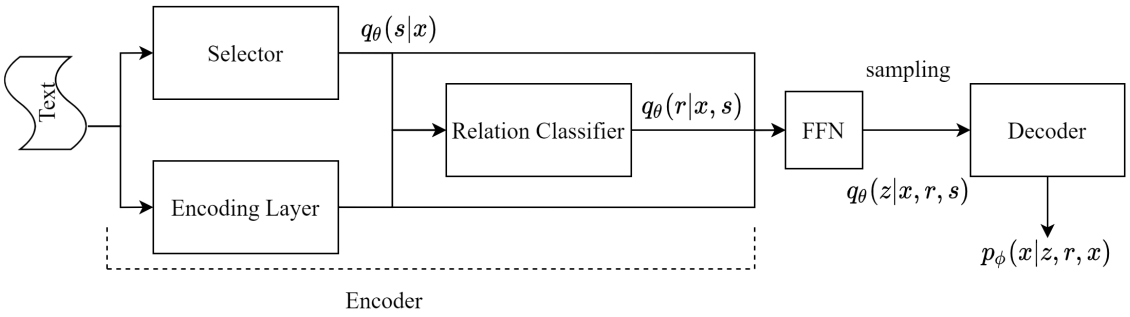

Figure 5.1: The architecture of SG-UREVA

As shown in Fig 5.1, we can use the same UREVA architecture to model the loss function Eq. 5.2. However, directly training the entire model is likely to cause unbalanced training result. For example, we already demonstrated the clustering performance of the UREVA model in the Section 4.4.4. The newly added selector component is likely to be ignored when training with this robust UREVA model, i.e., the selector will determine similar probabilities for all sentence groups since the UREVA model can be converged even without the selector component. This does not meet our expectations. As stated in the problem formulation, we need to know which sentences can be formed as a sentence group and what the relation of that group. The model cannot determine which sentences can be formed as a sentence group sin the selector is ignored. Therefore, we separate the selector from the entire model and iteratively train the selector and the remaining component, which we call VAE component in the following sections. That is, when the VAE component is trained, Selector will not be updated and $q_\theta(s|x)$ remains as a constant. In addition, we also propose a new loss function to train the Selector more effectively.

5.2.2 Loss Function of Selector

The role of selector is very similar to the role of SentDE proposed in the Section 3.3. Both of them adjust the output probability distribution to reduce the loss of the main model.

Therefore, the proposed loss function also has the same form as the reward in reinforcement learning. As shown in Eq. 5.3, the loss function can be defined as a reward function:

$$R = \sum_{i=1}^{K} q_\theta(s|x_i)(-\mathcal{L}_{x_i;VAE}) \tag{5.3}$$

Note that there is a negative sign in front of the loss value of the VAE component. This is to ensure that maximizing the reward is equivalent to reducing the loss of VAE. However, this loss function still cannot solve the problem of unbalanced training well. One of the main reasons is that in this loss function, we ignore to regularize the nature property of the probability distribution output by the selector. We believe that $q_\theta(s|x)$ output by a selector should have the following two important properties:

- The ratio of the number of correct sentence groups to the number of all sentence groups should be small. Because in a text, most of the randomly selected sentences form the sentence group, there should be no valid relation information.

- The probability distribution should be discriminative, i.e., 0.95 for the correct sentence group and 0.01 for the false

We use the combination of expectation and variance to regularize these two properties. Specifically, we stipulate that in each batch, the variance of all $q_\theta(s|x)$ should be larger, which addresses the second property. On the basis of satisfying the large variance requirement, we stipulate that the expectation needs to be close to a certain value. The reason is that large variance means that most of $q_\theta(s|x)$ are 0 or 1, so a certain expectation close to 0 means that the number of $q_\theta(s|x) = 1$ is much smaller than the number of $q_\theta(s|x) = 0$, which satisfies the first property. Combining these two regularizers and reward function Eq. 5.3, we can

define the final loss function of the selector in Eq. 5.4:

$$\mathcal{L} = \alpha||\mathbb{E}_{q_\theta(s|x)}[s] - \mathcal{E}||_2 - \beta Var(q_\theta(s|x)) - R \tag{5.4}$$

where $\mathcal{E}$ is the pre-defined expectation value and α and β are the parameters to control the weight of the two regularizers. In practice, $\mathcal{E}$ is chosen based on preliminary experiments on development set.

5.2.3 Architectures of Decoder, Encoder and Selector

Note that the proposed model has a structure similar to UREVA, so we also use the same approximation method to optimize the VAE loss function. One may refer to Section 4.3.2 to learn more about this approximate method. Based on this, the decoder architecture also uses convolutional neural network to score the input n-ary entities. Compared to the decoder of UREVA shown in Fig 4.4, the only difference of the decoder of SG-UREVA is that there are n entity embeddings concatenated in the first stage.

Since the encoder and selector have similar roles, that is, encode the input sentence group and output a probability, the two can be modeled with the same neural network architecture. In addition, since the model we proposed is to provide supervised signal to any relation classifier for training, technically we can use any supervised n-ary cross-sentence relation extraction model as the encoder and selector, such as the relation extractor proposed in the Section 3.2.

5.3 Experiment

5.3.1 Dataset

As a popular n-ary cross-sentence relation extraction evaluation dataset, PubMed dataset is used to evaluate the proposed model. The PubMed dataset is created by automatically labeling biomedical literature with Gene Drug Knowledge Database [37]. The detail information about this dataset can be found in Section 3.4.1.

5.3.2 Evaluation Metrics and Baselines

Both selector and VAE component are evaluated individually. In order to evaluate the VAE component, we use V-measure score to measure the cluster performance. V-measure [44] is the harmonic mean between homogeneity and completeness. We apply AGGCN as the encoder since it is the state-of-the-art supervised n-ary cross-sentence relation extraction model [17]. We also implement Simon's model as the baseline model [48]. Note that the original Simon's model uses PCNN as the encoder/relation classifier architecture. To make a fair comparison, we replace the PCNN with AGGCN.

We take the sentence groups that are labeled as "None" in the PubMed dataset as the noise instances. We observe whether the selector can distinguish between the "None" relation and other relations, i.e., the selector should output a probability close to 0 when the input sentence group is labeled with "None".

Table 5.1: The evaluation results of SG-UREVA's VAE component

	SG-UREVA	Simon's
V-measure	22.3	16.7

5.3.3 Experimental Results

Evaluation of VAE Component

The V-measure evaluation results of VAE component is shown in Table 5.1. The V-measure score of the proposed model is higher than that of Simon's. This proves that the performance of the proposed VAE-based unsupervised relation extraction model is still better than previous state-of-the-arts models under n-ary cross-sentence conditions. However, the V-measure score of the VAE component on the PubMed dataset is relatively low. We leave any additional error analysis to future work.

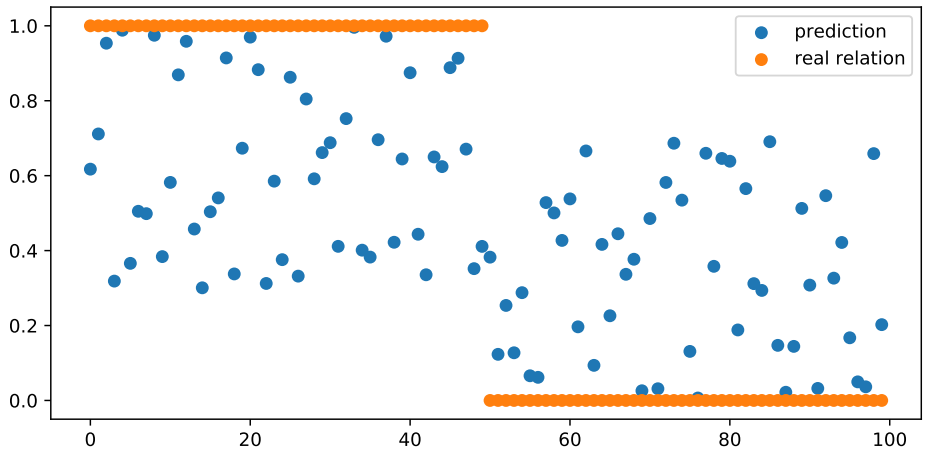

Figure 5.2: The evaluation results of SG-UREVA's selector

Evaluation of Selector Component

The evaluation results of selector component is shown in Fig. 5.2. We label the instances that are not marked as "None" in the PubMed dataset as 1, and the instances that are marked as "None" as 0. In this figure, the orange dots are the actual labeled results, and The blue points are the probabilities predicted by the selector. One can see that most of the blue points are roughly consistent with the distribution of orange points. This demonstrates that the selector can distinguish between the noise and the real sentence groups to some extent. However, some instances are incorrectly assigned. For example, in the area where the orange points are 0, the probabilities of some blue points are higher than 0.6. We think a possible reason is that the VAE component cannot perform well currently even on the noise-free dataset. Therefore, the selector that is iteratively trained with VAE cannot successfully distinguish all instances.

Chapter 6

Conclusions

6.1 Summary

In this work, we focused on the N-ary cross-sentence relation extraction problem. We first proposed a supervised n-ary cross-sentence relation extraction model based on reinforcement learning. This model can extract relationships from non-consecutive sentences and can automatically remove mislabeled instances in the dataset with the help of the proposed sentence distribution estimator.

After that, since there are few n-ary cross-sentence relation extraction datasets with labeled instances, it is difficult to train a complex supervised model. Therefore, we address unsupervised n-ary cross-sentence relation extraction problem. We first proposed two unsupervised binary intra-sentence relation extraction models, CURE and UREVA. Among them, CURE is based on the encoder:decoder architecture, where the encoder outputs relation information, which is used by the decoder to reconstruct the input dependency shortest path. The encoding information output by a trained encoder is then used to cluster sentences with similar relation information. UREVA uses a proabilistic model, which is variational autoencoder, to simulate this process. The encoder first treats the relation classification as an intermediate variable and assumes that the latent variable z is conditioned on the joint distribution of relation classification and the input sentence. After that, the decoder reconstructs the triplets that appear in the sentence based on the information sampled from z. Our experimental

results show that the performance of the UREVA model is better than that of the CURE model, so we use UREVA as the architecture to build an unsupervised n-ary cross-sentence relation extraction model.

Under the condition of n-ary cross-sentence, we assume that latent variable z is conditioned on three variables, input sentence group, the probability of that sentence group has relation, relation classification results. Similar to the sentence distribution estimator in supervised learning, we proposed a selector to perform iteratively training with the original VAE component. This selector that has the same architecture with encoder/relation classifier gives the probability of that whether the input sentence group contains a relation.

6.2 Future Work

N-ary Cross-sentence relation extraction is an important task and in this work we take the first steps towards addressing it under unsupervised setting, hoping that other researchers will follow. There are some directions that we will investigate in the future. The current supervised n-ary cross-sentence relation extractor can be complex and difficult to deploy. Therefore, we will investigate the possibility of reducing the number of model parameters without compromising model performance or the model's ability to encode non-consecutive sentences. The performance of the UREVA model on PubMed did not meet our expectations, even though its performance is better than state-of-the-arts models. We will explore the reasons for its current bottleneck to improve its performance on the n-ary cross-sentence relation extraction task.

Appendices

Appendix A

Appendix of Proposed Supervised Relation Extraction Model

A.1 Algorithm

The training procedure of the proposed model is shown in Algorithm 1:

A.2 Theoretical Analysis

We theoretically show that the proposed model can classify correctly labeled data and remove the incorrectly labeled data.

To formalize this statement, we first formally define the training data distribution. Suppose we have true distribution $p_X(x)$ and noisy distribution ξ. In general, ξ has zero mean. In addition, each sampled data of ξ is not correlated. Then $x_p \sim p_X(x)$ is the correctly labeled data and $x_n \sim p_X(x)\xi$ is the incorrectly labeled data. The main reason is that incorrectly labeled sentences have the same entity set with the correctly labeled sentences but the semantic information of incorrectly labeled sentences are shifted by some noise.

Then the statement is equivalent to corollary 1 below.

Algorithm 1: Model Training

Input: a set of sentence groups and relation label pairs
$\mathcal{H} = \{(g_1, r_1), (g_2, r_2), \cdots, (g_n, r_n)\}$

Output: The RE and SentDE trained on the input

Parameter: the number of training times for RE, SentDE and the whole model, M, J and K, respectively

Initialize parameters of the RE and SentDE model;
for $m = 1 \rightarrow M$ **do**
    RE receives $\mathcal{G} = \{g_1, g_2, \cdots, g_n\}$ as input
    RE outputs the classification results $\hat{\mathcal{R}} = \{\hat{r}_1, \hat{r}_1, \cdots, \hat{r}_n\}$
    calculate cross entropy loss based on the $\hat{\mathcal{R}}$ and $\mathcal{R} = \{r_1, r_2, \cdots, r_n\}$
    update parameters of RE model
end
for $k = 1 \rightarrow K$ **do**
    for $j = 1 \rightarrow J$ **do**
        SentDE samples instances $\mathcal{G}' = \{g'_1, g'_2, \cdots, g'_i\}$ from $\mathcal{G}$ via Eq. 3.6∼3.8
        do step 18∼19
        calculate reward based on the classification accuracy from RE via Eq. 3.9
        calculate policy gradient via Eq. 3.10
        update parameters of SentDE model via Eq. 3.11
    end
    SentDE samples instances $\mathcal{G}'$ from $\mathcal{G}$
    for $m = 1 \rightarrow M$ **do**
        RE receives the sampled instances $\mathcal{G}'$ as input
        RE outputs the classification results $\hat{\mathcal{R}}$
        calculate cross entropy loss based on the $\hat{\mathcal{R}}$ and $\mathcal{R}$
        update parameters of RE model
    end
end
return RE, SentDE

Corollary 1. After iteratively training SentDE and RE, we have:

$$P_{\theta,\gamma}(y=1|x_p) \gg P_{\theta,\gamma}(y=1|x_n) \tag{A.1}$$

where $P_{\theta,\gamma}(1|x_p)$ indicates the probability of selecting sampled x_p as the positive labeled data. To prove Corollary 1, we define two lemmas and give the proof sketch of these two lemmas:

Lemma 1. Let r be the average reward, for RL, we have

$$\begin{aligned} &\max R \equiv max(R_{\overline{p}} + R_{\overline{n}}) \leq max(R_{\overline{p}}) + max(R_{\overline{n}}) \\ &where\ R = \begin{cases} R_{\overline{p}} & R - r > 0 \\ R_{\overline{n}} & R - r < 0 \end{cases} \end{aligned} \tag{A.2}$$

Proof of Lemma 1. The objective of RL is to maximize the expected reward, as stated in Eq.A.3

$$\begin{aligned} R &= \sum_{\tau} R(\tau)\pi_{\theta,\gamma}(\tau) \\ &= \sum_{\tau} \big(R(\tau) - r\big)\, \pi_{\theta,\gamma}(\tau) \end{aligned} \tag{A.3}$$

Subtracting r from R is equivalent to the original reward function, because this is an unbiased estimation of expectation. Based on this, maximizing R is equivalent as follows:

$$\begin{aligned} &\max R \\ &\propto \max \left(\sum_{\tau \in \tau_p} \big(R_p(\tau)\big)\, \pi_{\theta,\gamma}(\tau) + \sum_{\tau \in \tau_n} \big(R_n(\tau)\big)\, \pi_{\theta,\gamma}(\tau) \right) \\ &\leq \max \left(\sum_{\tau \in \tau_p} \big(R_p(\tau)\big)\, \pi_{\theta,\gamma}(\tau) \right) + \max \left(\sum_{\tau \in \tau_n} \big(R_n(\tau)\big)\, \pi_{\theta,\gamma}(\tau) \right) \\ &where\ \tau_p \in \{\tau | R(\tau) - r > 0\}\ \tau_n \in \{\tau | R(\tau) - r < 0\} \end{aligned} \tag{A.4}$$

□

By maximizing the first term $R_{\overline{p}}$, Lemma 1 indicates that after training the RL, the data that has higher reward ($R-r>0$) will be assigned higher probability to be selected as truly labeled data. Similarly, the data that has lower reward ($R-r<0$) will be assigned lower probability by minimizing the second term $R_{\overline{n}}$.

Since the relation extraction task is a classification problem, we use cross-entropy as the loss function of the RE model[1]:

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_{x_p\sim p_X(x)}[ylog(\hat{y}(x)) + (1-y)log(1-\hat{y}(x))] \\ &+ \mathbb{E}_{x_n\sim p_X(x)\xi}[ylog(\hat{y}(x) + (1-y)log(1-\hat{y}(x))]\end{aligned} \tag{A.5}$$

Lemma 2. After training RE model, we have

$$\begin{aligned}&\min\mathcal{L} \equiv \min\mathcal{L}_r + \upsilon^2\mathcal{L}_n \\ \text{where}\ \ &\mathcal{L}_r = -\sum_{x\in\{x_p,x_n-\xi\}} ylog(\hat{y}(x)) + (1-\hat{y}(x))log(1-y_p)p(y|x)p(x) \\ &\int \xi_i\xi_j p(\xi)d\xi = \upsilon^2\delta_{ij} \\ &\mathcal{L}_n = \frac{1}{2}\sum_{x\in x_n-\xi}\frac{1}{\hat{y}(1-\hat{y})}(\frac{\partial\hat{y}}{\partial x})^2 p(x)\end{aligned} \tag{A.6}$$

$\mathcal{L}_n$ is positive definite and can be deemed as a regularization term when υ^2 is small. ξ_i and ξ_j are the noise vectors associated with input x_i and x_j

Lemma 2 indicates that the model will converge to the distribution of x_p with regularization. In other words, the cross-entropy loss of x_p is much lower than that of x_n.

Proof of Lemma 2. We expand the loss function as a Taylor series in powers of ξ and substitute the Taylor expansion into the loss function. Then the loss function can be re-

[1]We use binary classification for simplification, the case is the same in multi-class classification

written as:

$$\begin{aligned}\mathcal{L} &= \mathcal{L}_r + \upsilon^2 \mathcal{L}_e \\ \mathcal{L}_e &= \frac{1}{2} \sum_{x \in x_n - \xi} \left\{ \left[\frac{\hat{y} - y}{\hat{y}(1-y)} \right] \frac{\partial^2 \hat{y}}{\partial x^2} \right. \\ &\quad \left. + \left[\frac{1}{\hat{y}(1-\hat{y})} - \frac{(\hat{y}-y)(1-2\hat{y})}{\hat{y}^2(1-\hat{y})^2} \right] \left(\frac{\partial \hat{y}}{\partial x} \right)^2 \right\} p(y|x)p(x)\end{aligned} \tag{A.7}$$

where υ represents the amplitude of the noise, $\hat{y}$ represents the predicted relation label of x by the RE model and y represents the real label. In general, $\hat{y}$ represents the probability of predicting the correct label for x should be labeled as the correct relation. As proved in previous literature, the second and third term $\mathcal{L}_e$ vanish after training the model [3]. Then $\mathcal{L}_e$ is equivalent to:

$$\mathcal{L}_e = \frac{1}{2} \sum_{x \in x_n - \xi} \frac{1}{\hat{y}(1-\hat{y})} (\frac{\partial \hat{y}}{\partial x})^2 p(x) \tag{A.8}$$

Now $\mathcal{L}_e$ only has first derivatives and is positive definite. In other words, $\mathcal{L}_e = \mathcal{L}_n$ and it can be deemed as a regularizer. □

From Lemma 1 and Lemma 2: upon completion of the iterative training of the SentDE and RE models, the probability of selecting the correctly labeled data x_p as positively labeled data, will end up with a comparatively larger value than the incorrectly labeled data x_n. That is, corollary 1 is shown to be true.

Bibliography

[1] Amit Bagga and Breck Baldwin. Algorithms for scoring coreference chains. In *The first international conference on language resources and evaluation workshop on linguistics coreference*, volume 1, pages 563–566. Citeseer, 1998.

[2] Horace B Barlow. Unsupervised learning. *Neural computation*, 1(3):295–311, 1989.

[3] Chris M Bishop. Training with noise is equivalent to tikhonov regularization. *Neural computation*, 7(1):108–116, 1995.

[4] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[5] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250, 2008.

[6] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Audio chord recognition with recurrent neural networks. In *ISMIR*, pages 335–340. Citeseer, 2013.

[7] Razvan C Bunescu and Raymond J Mooney. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 724–731. Association for Computational Linguistics, 2005.

[8] Jinxiu Chen, Donghong Ji, Chew Lim Tan, and Zheng-Yu Niu. Unsupervised feature

selection for relation extraction. In *Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts*, 2005.

[9] Oier Lopez De Lacalle and Mirella Lapata. Unsupervised relation extraction with general domain knowledge. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 415–425, 2013.

[10] Hady Elsahar, Elena Demidova, Simon Gottschalk, Christophe Gravier, and Frederique Laforest. Unsupervised open relation extraction. In *European Semantic Web Conference*, pages 12–16. Springer, 2017.

[11] Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S Weld, and Alexander Yates. Unsupervised named-entity extraction from the web: An experimental study. *Artificial intelligence*, 165(1):91–134, 2005.

[12] Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. Reinforcement learning for relation classification from noisy data. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[13] Andrea Galassi, Marco Lippi, and Paolo Torroni. Attention in natural language processing. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[14] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610, 2005.

[15] Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. Hybrid speech recognition with deep bidirectional lstm. In *2013 IEEE workshop on automatic speech recognition and understanding*, pages 273–278. IEEE, 2013.

[16] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. Ieee, 2013.

[17] Zhijiang Guo, Yan Zhang, and Wei Lu. Attention guided graph convolutional networks for relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 241–251, 2019.

[18] Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd annual meeting on association for computational linguistics*, page 415. Association for Computational Linguistics, 2004.

[19] Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 33–38, 2010.

[20] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[21] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[22] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[23] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.

[24] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[25] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series.

[26] Hong Li, Sebastian Krause, Feiyu Xu, Andrea Moro, Hans Uszkoreit, and Roberto Navigli. Improvement of n-ary relation extraction by adding lexical semantics to distant-supervision rule learning. In *ICAART (2)*, pages 317–324, 2015.

[27] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2873–2879, 2016.

[28] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015.

[29] Angrosh Mandya, Danushka Bollegala, Frans Coenen, and Katie Atkinson. Combining long short term memory and convolutional neural network for cross-sentence n-ary relation extraction. In *Automated Knowledge Base Construction (AKBC)*, 2018.

[30] Diego Marcheggiani and Ivan Titov. Discrete-state variational autoencoders for joint discovery and factorization of relations. *Transactions of the Association for Computational Linguistics*, 4:231–244, 2016.

[31] Filipe Mesquita, Jordan Schmidek, and Denilson Barbosa. Effectiveness and efficiency of open relation extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 447–457, 2013.

[32] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*, 2010.

[33] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[34] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, 2009.

[35] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 809–816, 2011.

[36] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. PMLR, 2013.

[37] Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. Cross-sentence n-ary relation extraction with graph lstms. *Transactions of the Association for Computational Linguistics*, 5:101–115, 2017.

[38] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors

for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[39] Hoifung Poon and Pedro Domingos. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 1–10. Association for Computational Linguistics, 2009.

[40] Pengda Qin, Weiran Xu, and William Yang Wang. Robust distant supervision relation extraction via deep reinforcement learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2137–2147, 2018.

[41] Chris Quirk and Hoifung Poon. Distant supervision for relation extraction beyond the sentence boundary. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1171–1182, 2017.

[42] Thomas Rebele, Fabian Suchanek, Johannes Hoffart, Joanna Biega, Erdal Kuzey, and Gerhard Weikum. Yago: A multilingual knowledge base from wikipedia, wordnet, and geonames. In *International semantic web conference*, pages 177–185. Springer, 2016.

[43] Sebastian Riedel, Limin Yao, and Andrew McCallum. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer, 2010.

[44] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 410–420, 2007.

[45] Benjamin Rosenfeld and Ronen Feldman. Ures: an unsupervised web relation extraction system. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 667–674. Association for Computational Linguistics, 2006.

[46] Benjamin Roth, Tassilo Barth, Michael Wiegand, and Dietrich Klakow. A survey of noise reduction methods for distant supervision. In *Proceedings of the 2013 workshop on Automated knowledge base construction*, pages 73–78, 2013.

[47] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[48] Étienne Simon, Vincent Guigue, and Benjamin Piwowarski. Unsupervised information extraction: Regularizing discriminative approaches with relation distribution losses. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1378–1387, 2019.

[49] Alisa Smirnova and Philippe Cudré-Mauroux. Relation extraction using distant supervision: A survey. *ACM Computing Surveys (CSUR)*, 51(5):1–35, 2018.

[50] Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. N-ary relation extraction using graph-state lstm. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2226–2235, 2018.

[51] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[52] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy

gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[53] Thy Thy Tran, Phong Le, and Sophia Ananiadou. Revisiting unsupervised relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7498–7505, 2020.

[54] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[55] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[56] Denny Vrandečić. Wikidata: A new platform for collaborative data collection. In *Proceedings of the 21st international conference on world wide web*, pages 1063–1064, 2012.

[57] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. Explainable reasoning over knowledge graphs for recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5329–5336, 2019.

[58] Chenyan Xiong, Russell Power, and Jamie Callan. Explicit semantic ranking for academic search via knowledge graph embedding. In *Proceedings of the 26th international conference on world wide web*, pages 1271–1279, 2017.

[59] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9(4):611–629, 2018.

[60] Yulan Yan, Naoaki Okazaki, Yutaka Matsuo, Zhenglu Yang, and Mitsuru Ishizuka. Unsupervised relation extraction by mining wikipedia texts using information from the web. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1021–1029. Association for Computational Linguistics, 2009.

[61] Kaijia Yang, Liang He, Xinyu Dai, Shujian Huang, and Jiajun Chen. Exploiting noisy data in distant supervision relation classification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3216–3225, 2019.

[62] Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. Structured relation discovery using generative models. In *proceedings of the 2011 conference on empirical methods in natural language processing*, pages 1456–1466, 2011.

[63] Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1753–1762, 2015.

[64] Yuhao Zhang, Peng Qi, and Christopher D Manning. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2205–2215, 2018.

[65] Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander Smola, and Le Song. Variational reasoning for question answering with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[66] Di Zhao, Jian Wang, Yijia Zhang, Xin Wang, Hongfei Lin, and Zhihao Yang. Incor-

porating representation learning and multihead attention to improve biomedical cross-sentence n-ary relation extraction. *BMC bioinformatics*, 21(1):1–17, 2020.

[67] Michał Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. The united nations parallel corpus v1. 0. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3530–3534, 2016.