Unsupervised Relation Extraction: A Variational Autoencoder Approach

Chenhan Yuan and **Hoda Eldardiry**
Department of Computer Science, Virginia Tech
chenhan,hdardiry@vt.edu

Abstract

Unsupervised relation extraction works by clustering entity pairs that have the same relations in the text. Some existing variational autoencoder (VAE)-based approaches train the relation extraction model as an encoder that generates relation classifications. A decoder is trained along with the encoder to reconstruct the encoder input based on the encoder-generated relation classifications. These classifications are a latent variable so they are required to follow a pre-defined prior distribution which results in unstable training. We propose a VAE-based unsupervised relation extraction technique that overcomes this limitation by using the classifications as an intermediate variable instead of a latent variable. Specifically, classifications are conditioned on sentence input, while the latent variable is conditioned on both the classifications and the sentence input. This allows our model to connect the decoder with the encoder without putting restrictions on the classification distribution; which improves training stability. Our approach is evaluated on the NYT dataset and outperforms state-of-the-art methods.

1 Introduction

Relation Extraction (RE) methods aim to classify relations of entity pairs in a given sentence. RE is the basis of many NLP tasks, such as Information Extraction (Niklaus et al., 2018), Knowledge Graph Construction (Bosselut et al., 2019) and Information Retrieval (Liu et al., 2014). Conventional RE models classify relations based on preset rules (Zhou et al., 2005; Etzioni et al., 2008). These rules include syntactic patterns such as dependency structures and part of speech (POS). Although these rule-based methods achieve high accuracy, the rules can only be applied to limited types of sentences, which leads to application limitations (Wu et al., 2017). Recently, with the rise of deep learning, some work introduced deep neural networks, such as Piece-wise Convolutional Neural Network (PCNN) (Zeng et al., 2015), Graph Convolutional Networks (Guo et al., 2019), and Graph LSTM (Peng et al., 2017). These approaches perform well. However, they use supervised model training which requires large-scale labeled data which is not always available.

To reduce the reliance on labeled data, some work propose unsupervised relation extraction methods. Traditional unsupervised RE approaches compose vectors based on extracted sentence features. These feature vectors represent the relation information of sentences (Lin and Pantel, 2001; Yao et al., 2012). These methods then cluster sentences according to these feature vectors to classify sentences into relations. Even though these approaches achieve state-of-the-art performance, they heavily rely on hand-craft features and make many simplifying assumptions as mentioned in previous work (Marcheggiani and Titov, 2016). Other unsupervised RE techniques follow a discriminative approach to train more expressive models. These techniques can provide a sufficient learning signal to train deep neural networks as relation classifiers. For example, some work achieved a discriminative approach using a variational autoencoder (VAE)-based method (Marcheggiani and Titov, 2016). The main idea of this approach is to train an encoder as a relation classifier, which converts input sentences into relation classifications. A decoder then reconstructs entity pairs $(head\_entity, tail\_entity)$ of input sentences according to the classifications. The decoder's reconstruction loss provides the encoder with a supervision signal so the encoder can be implemented as a deep neural network.

While this approach trains the relation classifier with supervision, the model has been shown to be unstable when tested on the New York Times-Freebase (NYT-FB) dataset (Riedel et al., 2010). Specifically, the model always results in one of two training results depending on the hyper-parameter

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 1929–1938
November 7–11, 2021.

setting: the model either predicts the same relation for all input sentences, or the prediction probability of each relation is always the same (Simon et al., 2019). To improve model stability, recent work proposed adding two regularization losses to the original objective function (Simon et al., 2019). Adding these two extra losses skews the prediction probability distribution of the relation of one sentence while pushing the relation distribution of a set of sentences to be uniform. However, during our experimental implementation of this model, we note that even with adding the regularization terms, there is still an unresolved problem that may negatively impact performance.

The key problem of this VAE-based model is that the relation classification result r is a latent variable. This leads to the following two limitations. 1) The decoder can not follow the traditional VAE training process of directly sampling from r as it would result in interrupting the gradient passing. Therefore, the decoder is not directly connected to the encoder and can not update it in a timely manner. 2) The additional regularization loss terms serve the role of the KL divergence term in the VAE loss function. This requires the classification to follow a pre-defined prior distribution, namely, a uniform distribution. These two limitations may lead to unbalanced VAE training of the relation extraction encoder model and decoder. Consequently, the VAE loss converges, while the relation classification accuracy is not relatively high.

We propose a **V**ariational **A**utoencoder-based **U**nsupervised **R**elation **E**xtraction model (UREVA) which overcomes the aforementioned limitations. In particular, our proposed approach treats the relation classification r as an intermediate variable conditioned on the input sentence x, while the latent variable z is conditioned on the joint distribution of x and r. The decoder uses r and data sampled from z to reconstruct entity pair in the input sentence. Through the sampling process, the decoder and the latent space establish a connection. This addresses limitation 1 mentioned above. In addition, since r is no longer a latent variable, fitting the latent variable to any prior distribution does not affect the classification results. This overcomes limitation 2.

We run experiments on the NYT-FB (Riedel et al., 2010) and SemEval (Hendrickx et al., 2010) datasets to compare the performance of our proposed method against other state-of-the-art methods. Experimental results show that our model generates more accurate classification results than state-of-the-art methods on both datasets. Our key contributions can be summarized as follows:

- We propose UREVA; a VAE-based unsupervised relation extractor that connects decoder with encoder without putting restrictions on the distribution of relation classification.
- We show that compared with previous works, UREVA learns the relation classification instead of predicting uniform classification results.
- We show that UREVA outperforms state-of-the-art methods on the NYT-FB dataset. We also demonstrate UREVA's performance on a new dataset, SemEval, which has not been explored by previous work.

2 Variational Autoencoder Background

In this section, we review the variational autoencoder (VAE) architecture and its objective function. VAE is a probabilistic generative model that describes an observation in latent space. The goal of the VAE is to train a decoder via the joint distribution $p_\phi(x, z) = p_\phi(z)p_\phi(x|z)$, where $p_\phi(z)$ is a prior distribution of latent variable z and the posterior distribution $p_\phi(x|z)$ is the decoder that generates x given z. In general, since computing the true posterior distribution $p_\phi(z|x)$ is intractable, an encoder $q_\theta(z|x)$ is used to approximate it. The objective of the VAE is to minimize the KL divergence of the encoder $q_\theta(z|x)$ and the decoder $p_\phi(z|x)$ such that the two distributions are similar. Expanding $p_\phi(x)$, it can be expressed as follows:

$$\begin{aligned} log(p_\phi(x)) &= \mathbb{E}_{q_\theta(z|x)}[log(p_\phi(x))] \\ &= KL(q_\theta(z|x)||p_\phi(z|x)) + \mathcal{L}(\theta, \phi; x) \end{aligned} \tag{1}$$

We have the following inequality since $KL(\cdot) \geq 0$:

$$\begin{aligned} &log(p_\phi(x)) \geq \mathcal{L}(\theta, \phi; x) \\ &= \mathbb{E}_{q_\theta(z|x)}[log(p_\phi(x|z)) - log(q_\theta(z|x))] \\ &= \mathbb{E}_{q_\theta(z|x)}[log(p_\phi(x|z))] - KL(q_\theta(z|x)||p_\phi(z)) \end{aligned} \tag{2}$$

3 Proposed UREVA Method

The goal of our model is to extract a relation between an entity pair in a given sentence. The open question is how to train a relation extraction model

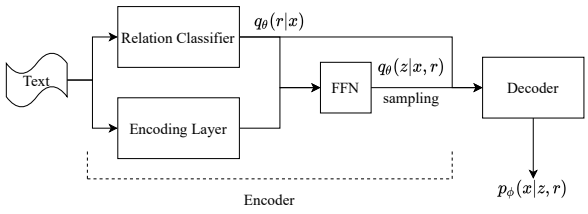

Figure 1: Proposed model architecture. FFN: two-layer feed-forward network.

without labeled data. In order to provide a supervision signal to the relation extractor, the initial idea is to use an encoder-decoder architecture. In this architecture, the relation extractor is an encoder that outputs relation classifications. In addition, a decoder reconstructs the encoder input, i.e., the sentence, given the classifications. The two models are jointly trained and the only way to reduce reconstruction loss is to let the encoder compress all the relation information to the classification results. In this way, the classifications become the bottleneck of the whole architecture.

In practice, we choose VAE instead of a general autoencoder to implement this because classifications are probabilistic. Specifically, as shown in Fig 1, we divide the encoder network into two parts: the relation classifier, and the encoding layer. The relation classifier calculates the probability $q_\theta(r|x)$ that a given input sentence x is classified into relation r. We let the latent variable z (representing sentence and relation information) be conditioned on x and r, which can be written as $q_\theta(z|x,r)$. We use the encoding layer to model the relation probability $q_\theta(z|x,r)$. The decoder network then reconstructs the probability $p_\phi(x|r,z)$ of input sentence x given samples of z and relations r. Finally, as shown in Eq. 3, our model has the following mathematical property:

$$q_\theta(z,r|x) = q_\theta(r|x)q_\theta(z|x,r) \tag{3}$$

Note that the relation extractor is not used as the entire encoder. We explain this design choice in Sec 3.5.

3.1 VAE-based Objective Function

Based on the proposed idea, the objective function of our model can be derived from the original VAE objective function by substituting r as follows:

$$log(p_\phi(x,r)) \geq \mathbb{E}_{q_\theta(z,r|x)}[log(p_\phi(x|r,z))] - KL(q_\theta(z,r|x)||p_\phi(z)) \tag{4}$$

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 as the objective function. We then substitute Eq. 3 into the objective function. This enables us to rewrite the objective function as shown in Eq. 5:

$$\begin{aligned}\mathcal{L} = \mathbb{E}_{q_\theta(r|x)}[\mathbb{E}_{q_\theta(z|x,r)}[logp_\phi(x|r,z)]+ \\ KL(q_\theta(z|x,r)||p_\phi(z))]+ \\ \sum_r \sum_z q_\theta(z|r,x)\mathcal{H}(q_\theta(r|x))\end{aligned} \tag{5}$$

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

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.

3.2.1 Decoding Approximation

As seen in Eq. 5, the goal of the decoder is to reconstruct the input sentence given relation classification r and latent variable z and compute the probability $logp_\phi(x|r,z)$. 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, instead of reconstructing the input sentence, the decoder is required to reconstruct the entity pair in the input sentence and compute the probability $logp_\phi(e_{head}, e_{tail}|r,z)$. This entity pair probability can be computed via Eq. 6 as follows:

$$p_\phi(e_{head}, e_{tail}|r,z) = \frac{\phi(e_{head}, e_{tail}|r,z)}{\sum_{i,j \in E; i \neq j} \phi(e_i, e_j|r,z)} \tag{6}$$

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

However, the calculation of the denominator of Eq. 6 is computationally intensive. Following (Simon et al., 2019), 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. 7, $p_\phi(e_{head}, e_{tail}|r, z)$ is equivalent to:

$$\begin{aligned} & p_\phi(e_{head}, e_{tail}|r, z) \propto \\ & -log(\sigma(\phi(e_{head}, e_{tail}|r, z))) \\ & -\sum_k \mathbb{E}_{e_i \in E}[log(\sigma(-\phi(e_{head}, e_i|r, z)))] \\ & -\sum_k \mathbb{E}_{e_i \in E}[log(\sigma(-\phi(e_j, e_{tail}|r, z)))] \end{aligned} \quad (7)$$

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.

3.2.2 Encoding Approximation

To enable VAE to pass the gradient during random sampling, a common method is to use reparameterization to simulate $q_\theta(z|r, x)$ (Kingma and Welling, 2013). 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 follows $\mathcal{N} \sim (0, 1)$.

By utilizing this trick, we can calculate the KL divergence term in Eq. 5. 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 weights of the entropy of different r's into a constant.

3.3 Encoder Architecture

According to previous work, entity types are the most significant features that represent relation information of sentences (Tran et al., 2020). We follow this work and apply two different feed-forward networks to encode entity types as the relation classifier and encoding layer.

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

$$\begin{aligned} r &= W_r^T(t_h t_t) + b_r \\ \hat{x} &= W_e^T(t_h t_t) + b_e \\ z &= W_{en}^T(\hat{x} \oplus r) + b_{en} \\ \hat{z} &= RT(z, \epsilon) \end{aligned} \quad (8)$$

where $\hat{z}$ 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.

3.4 Decoder Architecture

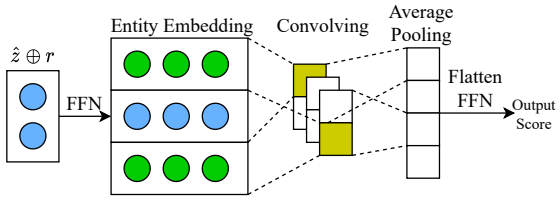

Figure 2: The decoder architecture

we use a CNN to reconstruct the entity pair. Note that previous work used RESCAL, which is a three-way tensor decomposition method and builds a three-dimensional matrix to represent the relation embedding (Nickel et al., 2011). However, our decoder needs to sample from the $q_\theta(z|r, x)$ distribution to establish a connection with the encoder. And the dimension of the sampling result of z is much smaller than the dimension of a three-way 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 entity pair according to that term. As shown in Fig 2, 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. Decoder should learn this embedding matrix along with CNN parameters. CNN uses n_f filters, each of kernel size $\mathbb{R}^{2 \times n_d}$, to extract features from c_{in}. Then 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, i.e., the score $\phi(\cdot)$ in Eq. 7, via a linear layer.

3.5 Key Insights

If we directly use encoder as the relation classifier as previous work did and let the classifier output a relation classification r (Marcheggiani and Titov, 2016), then the encoder can be expressed as $q_\theta(r|x)$. We keep other parts unchanged, i.e., the decoder reconstructs the encoder input based on r. Finally, the loss function can be defined via Eq. 9

$$\mathcal{L} = \mathbb{E}_{q_\theta(r|x)}[logp_\phi(x|r)] + KL(q_\theta(r|x)||p_\phi(r)) \quad (9)$$

Previous work approximates Eq. 9 by applying Eq. 10 as follows:

$$\mathcal{L} = \mathbb{E}_{q_\theta(r|x)}[logp_\phi(e_i|e_{-i}, r)] + \mathcal{H}(q_\theta(r|x)) \quad (10)$$

where e_i is the predicted entity and e_{-i} is the given entity and $\mathcal{H}(\cdot)$ is the entropy function. The model reconstructs the entity pair of the input sentence by predicting the missing entity. Comparing Eq. 10 with the VAE loss in Eq.2, r in Eq. 10 is the latent variable z in the VAE loss. In addition, the defined entropy of r is equivalent to the KL divergence of the prior probability $p_\phi(r|x)$ with uniform distribution. This leads to two outcomes: (1) The decoder cannot sample from the latent space r because the sampling process interrupts the gradient passing process. In addition, the same reparameterization trick in Sec 3.2.2 cannot be applied to solve this problem (i.e., sampling the decoder from r) because r is not a Gaussian distribution. Therefore, the decoder is not directly connected to the encoder and can not update it in a timely manner. (2) As the general VAE methods predefine z as a normal distribution, the entropy of r pushes the prior probability $p_\phi(r|x)$ to a uniform prior distribution. These two outcomes may lead to unbalanced training of the encoder and decoder. That is, after many training iterations, the reconstruction loss converges, which implies that the decoder is well-trained, while the encoder outputs a uniform distribution for sentence x (the probability of classifying x as any relation is the same). This answers the instability question raised in previous work (Simon et al., 2019).

We incorporate the relation classifier as a component of the encoder instead of using it as the encoder to avoid these two limitations. Unlike previous works, we regard r as an intermediate variable instead of a latent variable and r is conditioned on the input sentence x. Our proposed model is therefore guaranteed to be more stable. Next, we list two guarantees of model stability. 1) According to Eq. 3, if the encoder can map the input x to the latent space z, it must learn a good classifier $p_\theta(r|x)$, because this is an essential step for the mapping process. 2) Our decoder $q_\phi(x|z,r)$ reconstructs entity pairs based on the sampled z and r, while the decoder of the previous model does not receive any information from the encoder. This ensures that the proposed decoder utilizes the information from the sampled data. Therefore, the relation classifier can receive the gradients to update.

4 Experiments

4.1 Dataset and Evaluation Metrics

Data: Following previous work (Tran et al., 2020), 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 (Bollacker et al., 2008), 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 mentioned in previous work (Tran et al., 2020), 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 (Hendrickx et al., 2010). 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. We use 20% of these sentences as the test set.

Evaluation Metrics: B-cube (B^3) (Bagga and Baldwin, 1998), V-measure (Rosenberg and Hirschberg, 2007) and Adjusted Rand Index (ARI) (Hubert and Arabie, 1985) 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.

4.2 Models

4.2.1 UREVA Hyperparameter Settings

The hyperparameters are set based on experiments on the development set with manual tuning. The output dimension of the encoding layer is 64, and the sample size of z is also 64. The entity embedding dimension of decoder is set to 10, which is the same as the output dimension of the sampled result mapping layer of the decoder. The constant c is set to 0.01 and the number of CNN filters is 40. We use Adam with 0.005 learning rate to optimize the

[1]The detailed preprocessing steps are described in previous work (Marcheggiani and Titov, 2016).

# r	model	B^3	V-measure	ARI
10	RelLDA	29.1	30.1	13.2
	March(L_{sd})	38.7	37.5	26.1
	Simon	32.6	30.5	23.8
	EType+	41.9	40.6	30.7
	UREVA	**43.1**	**42.0**	**31.6**
16	RelLDA	27.8	29.4	12.2
	March(L_{sd})	36.9	37.4	28.1
	Simon	30.7	29.8	23.6
	EType+	41.5	41.3	30.5
	UREVA	**43.4**	**42.3**	**31.5**
100	RelLDA	25.1	27.9	9.7
	March(L_{sd})	35.1	26.2	17.1
	Simon	29.6	27.3	16.8
	EType+	35.8	35.4	27.3
	UREVA	**41.9**	**43.2**	**29.7**

Table 1: The evaluation results of UREVA and baseline methods on NYT-FB dataset. Note that # r indicates that how many clusters in each model.

proposed model. The batch size is set to 100.

4.2.2 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 (Yao et al., 2011) is designed for relation discovery task based on LDA model (Blei et al., 2003). The topic distribution is replaced by relation distribution, modeled by Dirichlet prior distribution. 2: March (Marcheggiani and Titov, 2016) is a VAE-like model, which encodes relation classification and train a decoder to reconstruct the entity pair 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_{sd}). 3: Simon (Simon et al., 2019) 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. 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 (Tran et al., 2020).

4.3 Experimental Results

4.3.1 Clustering Results

Performance on NYT-FB: Table 1 shows the average results across three-runs of each model. We follow previous work that set the number of relation clusters as $r = 10, 16, 100$ (Tran et al., 2020). The performance of UREVA on NYT-FB dataset is better than that of state-of-the-art models. In general, considering three evaluation metrics and three different number of relation clusters settings, the performance ranking of the model is UREVA>EType+>Simon>March(L_{sd})>RelLDA. Another thing to note is that the performance of all baselines drops significantly when $r = 100$, while the performance of UREVA remains roughly the same as $r = 10, 16$. For example, the V-measure score of EType+ is 41.3 when $r = 16$, while the V-measure score is 35.4 when $r = 100$. The V-measure score of EType+ drops 14%, while that of UREVA increases. This is expected because the more relation clusters predicted by the model, the more likely it is to fit the true relation distribution.

# r	Model	B^3	V-measure	ARI
5	Simon	22.3	11.2	9.7
	UREVA	**24.5**	**13.8**	**11.7**
9	Simon	21.6	11.5	10.6
	UREVA	**25.1**	**14.4**	**12.1**

Table 2: The evaluation results of UREVA and baseline methods on SemEval dataset. Note that # r indicates that how many clusters in each model.

Performance on SemEval: We also report the performance of UREVA and Simon models on the SemEval dataset in Table 2. Note that in experiments, we found that most of the entities labeled in the SemEval dataset are not named entities. And the way this dataset annotates entities can leak relation information to the models. That is, all models based on pre-defined features, such as entity types and dependency path, cannot be evaluated on this dataset. However, We argue that it is good enough to use this dataset to compare the performance of UREVA with only the Simon model because Simon is the state-of-the-art among VAE-based models and the goal of our proposed model is to improve the performance of the VAE-based model.

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 re-

placed the encoder architecture with PCNN, which is the same used in the Simon model. In addition, since there are only 10 different relations in the SemEval dataset, including "Others", we set the number of clusters of the models to 5 and 9.

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 that 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. Moreover, compared with Simon, UREVA can maintain a relatively high classification accuracy on SemEval.

4.3.2 Analysis of Classification Accuracy

As indicated in Section 4.3.1, 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 (Simon et al., 2019; Tran et al., 2020). 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.

As shown in Figure 3, 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, we list the relation distribution output by the model

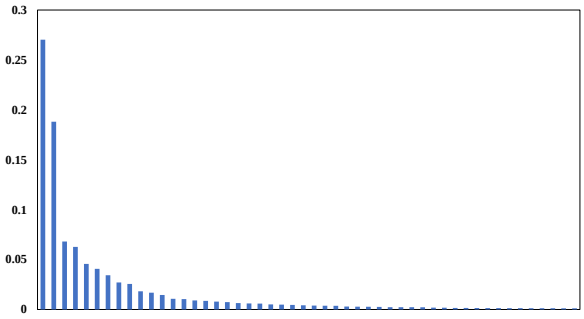

Figure 3: The real relation groups. The ordinate represents the percentage of the number of sentences in each relation group to the number of sentences in the dataset. The x-axis is the relations sorted according to the number of sentences contained. For ease of observation, the x-axis label is omitted

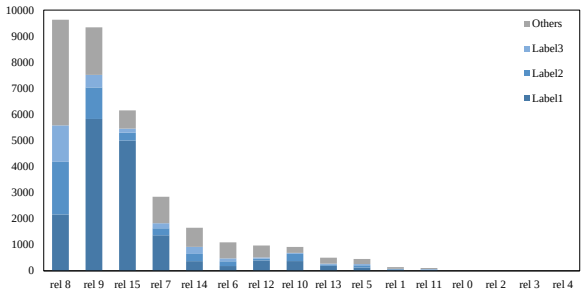

Figure 4: Predicted relation groups. rel_i is i-th predicted relation group. Label1, Label2, Label3 are the real relation that appears the top, second, and third most frequently in a relation group, respectively. The ordinate represents the number of sentences classified into each relation group.

on the test set and the number of different relations output by the model is set to 16. 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, supposing 50 sentences in the test set are predicted to be relation r_p0. Among these 50 sentences, 16 sentences are labeled as actual relation r_r1, 15 sentences are labeled as r_r2, 14 sentences are labeled as r_r3, and 5 sentences are labeled as r_r4. Then r_r1, r_r2 and r_r3 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.

Next, we provide a qualitative analysis, which

shows that the different relation classes predicted by UREVA have different meanings. As shown in Table 3, 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.

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

Table 3: Example of meanings of top-frequency real relations of each predicted relation group.

5 Related Work

Unsupervised RE uses unlabeled data to train a model that clusters sentences with the same relation information together. Initially, some work proposed using some linguistic patterns extracted from sentences as the contexts of entity pairs (Hasegawa et al., 2004). Once these contexts are vectorized, the model can cluster sentences with similar contexts. Follow-up work improved model performance by reducing the frequency of noisy words (Chen et al., 2005). Inspired by this approach, some work combined linguistic patterns and surface context to design a two-step clustering algorithm (Yan et al., 2009).

Some methods deal with unsupervised RE from the perspective of probabilistic generative models. Rel-LDA introduced the Latent Dirichlet Allocation (LDA) approach to unsupervised RE (Yao et al., 2011; Blei et al., 2003). In this method, the topic distribution is replaced by the relation distribution and Expectation Maximization algorithm is applied to cluster similar relations. Some work extended this model for general domain knowledge (De Lacalle and Lapata, 2013). They applied First Order Logic rules to extract patterns from knowledge base such that the model can be enhanced by using patterns as the prior knowledge.

Some work used the idea of autoencoder to provide supervised signal to RE. The CURE used an encoder to extract relation information from sentences (Yuan et al., 2020). The decoder reconstructs the dependency path between the entity pairs based on the relation information. After training, the relation information output by the encoder is used to cluster sentences with similar relations.

However, some work proposed to let the encoder output the classification result instead of an information vector (Marcheggiani and Titov, 2016). In this method, the decoder reconstructs the entity pair 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. More recently, some work considered that this model is not stable (Simon et al., 2019). This is because the model only predicts one same relation for all input sentences, or 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. This loss term requires that all relation classification results of the model conform to the uniform distribution. The PCNN architecture in this model is then replaced by entity type feature, since entity type was considered as the most important feature (Tran et al., 2020).

6 Conclusion

In this paper, we present UREVA, a variational autoencoder-based unsupervised relation extraction model. We consider relation classification as an intermediate variable to solve the model training instability problem that appeared in previous works. In UREVA, the classification is conditioned on the input sentence, and the latent variable is conditioned on the the joint distribution of the classification and the input sentence. Then the decoder reconstructs the entity pair in the input sentence by sampling from the latent space and the relation classification. Experiments on two public datasets show that UREVA outperforms state-of-the-art models.

References

Amit Bagga and Breck Baldwin. 1998. 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.

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

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. 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.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Jinxiu Chen, Donghong Ji, Chew Lim Tan, and Zheng-Yu Niu. 2005. Unsupervised feature selection for relation extraction. In *Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts*.

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

Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S Weld. 2008. Open information extraction from the web. *Communications of the ACM*, 51(12):68–74.

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

Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. 2004. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 415–422, Barcelona, Spain.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. 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.

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

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

Dekang Lin and Patrick Pantel. 2001. Dirt@ sbt@ discovery of inference rules from text. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 323–328.

Xitong Liu, Fei Chen, Hui Fang, and Min Wang. 2014. Exploiting entity relationship for query expansion in enterprise search. *Information retrieval*, 17(3):265–294.

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

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. 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.

Christina Niklaus, Matthias Cetto, André Freitas, and Siegfried Handschuh. 2018. A survey on open information extraction. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3866–3878, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

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

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

Andrew Rosenberg and Julia Hirschberg. 2007. 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, Prague, Czech Republic. Association for Computational Linguistics.

Étienne Simon, Vincent Guigue, and Benjamin Piwowarski. 2019. 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, Florence, Italy. Association for Computational Linguistics.

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

Yi Wu, David Bamman, and Stuart Russell. 2017. Adversarial training for relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1778–1783, Copenhagen, Denmark. Association for Computational Linguistics.

Yulan Yan, Naoaki Okazaki, Yutaka Matsuo, Zhenglu Yang, and Mitsuru Ishizuka. 2009. 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*, pages 1021–1029, Suntec, Singapore. Association for Computational Linguistics.

Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. 2011. Structured relation discovery using generative models. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1456–1466, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Limin Yao, Sebastian Riedel, and Andrew McCallum. 2012. Unsupervised relation discovery with sense disambiguation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 712–720, Jeju Island, Korea. Association for Computational Linguistics.

Chenhan Yuan, Ryan Rossi, Andrew Katz, and Hoda Eldardiry. 2020. Clustering-based unsupervised generative relation extraction. *arXiv preprint arXiv:2009.12681*.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. 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, Lisbon, Portugal. Association for Computational Linguistics.

GuoDong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 427–434, Ann Arbor, Michigan. Association for Computational Linguistics.