# A Experimental Setup

## A.1 Evaluation

For the NYT dataset, we adopt the standard micro F1 score, recall (R) and precision (P) as the metrics for the NER and RE subtasks. For the final result, a correct prediction is that the extracted triple matches the ground truth including two entities, relation direction and relation type. For the NER subtask, we consider the entity type, length, and position in sentences. For the SemEval-2018 dataset, we use the official script which calculates the macro-average F1 score.

## A.2 Training and Hyperparameters

The input word is projected to a 200-D pre-trained GloVe (Pennington et al., 2014) word embedding. The hidden state of the encoder is 300-D. The dimension of GRNN units (including the LSTM, GRU and RCNN units) is 200-D. We set $\lambda = 2.0$. For regularization we apply dropout with $p = 0.5$. To select better models, we divide the NYT training data into 100 pieces and the training data of SemEval-2018 into four pieces. We set $k = 1$ and $k = 6$ on SemEval-2018 and NYT dataset respectively. We first use single-sample training to get a good model and then adopt the batch (64) training to fine tune the model. We use the Adam optimization algorithm to update model parameters. The initial learning rate is 0.001 and the decay rate is 0.9. We ran the experiments on an Intel(R) Xeon(R) CPU E7-4830 v3 @ 2.10GHz (Mem: 976G) and one GPU Tesla K40c. The training time for NYT and SemEval-2018 datasets is 3h/epoch and 0.1h/epoch respectively.

# B Training Algorithm

We describe the inference and feedback details of Algorithm 1 as follows:

(1) In the inference step, this model first predicts/infers the label sequence, as shown in line 1, and uses the combination of the predicted entities as triple candidates, as shown in line 2-3. Then, the model dynamically extracts subgraphs of multiple triples to construct the forest on the top layer, as shown in line 5-9. This model uses the indices of $s$ and $o$ to find the LCA index as $p_w$. Then, the model will map the above indices into the GRNN to get the vector representations for a triple as shown in line 8. In line 10-16, this model also memorizes the unexpected entity pairs that are not predicted, using the same operations of line 5-9.

(2) In the feedback step, this model generates the relation type assumption for each predicted triple by using a single layer neural network, as shown in line 17. Then, it compares the assumptions with the ground truth to get the feedback signal to update the model state through backpropagation. The feedback signal is generated by the loss function to calculate the gradients.

# C Ablation Study

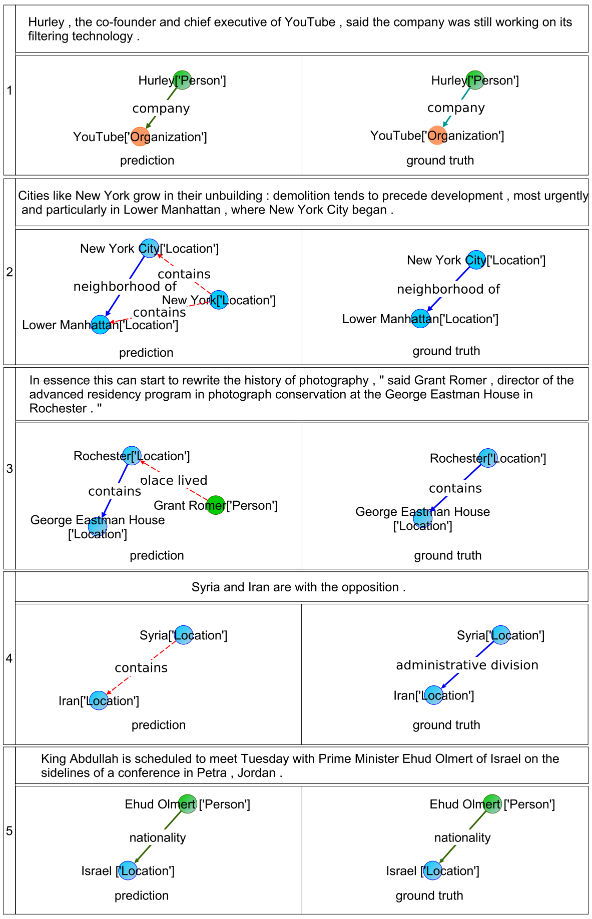
Table 1 shows an ablation study of the effects of multi-task training and pipeline training. Two system denotes a fine-tuned NER system and a GRNN system. Here we use an independent GRNN system as NER system. A first observation is that a fine-tuned NER system can improve the final results by 4.77%. Because it reduces the unknowns in the testing stage by explicitly indicate the entities. The draw back is that training is a multi-stage pipeline. We need to first fine tune an NER system and then predict the NER labels $\hat{Y}_{ner}$. Prediction results are also written to disk. Then we train the joint entity and relation extraction model that can use the $\hat{Y}_{ner}$ information as a supervisor.

Removing the multi-task training loss degrade the performance. This means multi-task training makes the RE subtask benefit from the NER subtask. When we remove the NER system, the performance drop significantly (6.19%). This means the NER is an important componment for improving the relation extraction process. Then, we remove NER system and multi-task training so that we first train the lower layer NER subtask and then freeze the parameters of shared layer. and then we train the RE subtask. The result also decreases because the lower layer represenation cannot encode the information from relation extraction subtask to interact with NER subtask. This means the multitask joint training is essential for subtask interaction. When we do not freeze the lower layer, the result also drops. Because when we train the RE subtask, the memory about NER subtask will continue to decrease. The drop of NER performance decides the low result RE subtask.

We also apply our inference training pipeline to BERT which is a pre-trained bidirecftional transformer and have proven very effective in many NLP tasks. BERT has hundreds of million parameters and trained on large scale corpus. We use fine-tuning based method to train a simple BERT model for TE. Table 2 shows the result. BERT achieves a slightly higher result than our GRNN model. Tis is because the NER result of this model is higher. The drawback is that the computational cost. When we remove the multi-task, NER system and Frozen NER, this model almost forget all the NER memory, so the RE subtask is failed. Little fine-tuning can achieve different functions which also means BERT is sensitive to little parameter change.

## D Case Study

As a case study for evaluating how the GRNN classify the relations, we pick some samples from NYT dataset and analyze the result of Bi-GRNN-LSTM model.



Hurley , the co-founder and chief executive of YouTube , said the company was still working on its filtering technology .

Cities like New York grow in their unbuilding : demolition tends to precede development , most urgently and particularly in Lower Manhattan , where New York City began .

In essence this can start to rewrite the history of photography , " said Grant Romer , director of the advanced residency program in photograph conservation at the George Eastman House in Rochester . "

Syria and Iran are with the opposition .

King Abdullah is scheduled to meet Tuesday with Prime Minister Ehud Olmert of Israel on the sidelines of a conference in Petra , Jordan .

Figure 1: Some extracted triples where the left side is the prediction results and the right side is the ground truth, and the red edge denotes the inconsistent predictions, and the other colors are the consistent predictions

Figure 1 shows the results. In the first sample, our model extracts the triple (*Hurley, company, YouTube*) since the *Hurley* and *YouTube* share the relation vertex "*co-founder*" in the dependency graph. This relation vertex implies the *"/business/person/company"* relation.

In the second sample, our model extracts three triples. For (*New York City, neighborhood of, Lower Manhattan* ), in the dependency graph, the vertex "*began*" appears in the path of *New York City* to *Lower Manhattan*. "*began*" might generate the relation type "*neighborhood of*" when two entities are locations. One mistake is (*New York, contains, New York City*) which means *"New York"* contains its own. Two entities share the same relation vertex "*grow*" in the dependency path *New York* ← *Cities* ← *grow* → *tends* → *urgently* → *Manhattan* → *began* → *New York City*. This relation vertex leads to the relation "*contains*". For the left triple, the situation is similar to the above description. This means literally considering the relationship might cause a misleading. Understanding complex spatial relationships needs to be further improved.

For the third sentence *"George Eastman House in Rochester"* implies that (*Rochester, contains, George Eastman House*). *" Grant Romer ... at George Eastman House in Rochester"* only indicates a state but doesn't mean the relation type *"place lived"*. Understanding time relationships needs to be further improved.

For the last sentence, it detects the entity mentions but not the relation type. Two entities share the same relation vertex "*opposition*" so the predicted (*Syria, contains, Iran*) relation might be a coincidence. Because the training data is generated by distant supervsion technique, some samples may produce noise. This also indicate that we need to reduce the noise of training data.

Table 2 shows the class-aware prediction result. We observe "*nationality*", "*company*", "place_lived" and "contains" relations are easier to extract. "*neighborhood_of*" relation only achieve a 25% F1 score. Other relations in the test set are not extracted. This might be caused by the relation imbalance problem in the training data.

## E Vertex Representation Analysis

To analyze the vertex embeddings learned for relation classification, Figure **??** shows the visualization of vertex embeddings distribution. visualize the also imply the semantic relationship between relation types. We pick some samples from

Table 1

| Model | Precision | Recall | F1 |
|---|---|---|---|
| Two system + multi-task training | 59.63 | 57.97 | 58.79 |
| –multi-task | 52.91 | 55.18 | 54.02 |
| –NER system | 54.9 | 50.5 | 52.6 |
| –multi-task, NER system | 49.61 | 49.11 | 49.36 |
| –multi-task, NER system, Frozen NER | 38.87 | 33.16 | 35.79 |

Table 2

| Model | Precision | Recall | F1 |
|---|---|---|---|
| Two system + multi-task training | | | |
| –multi-task | | | |
| –NER system | 50.68 | 56.20 | 53.30 |
| –multi-task, NER system | | | |
| –multi-task, NER system, Frozen NER | – | – | – |

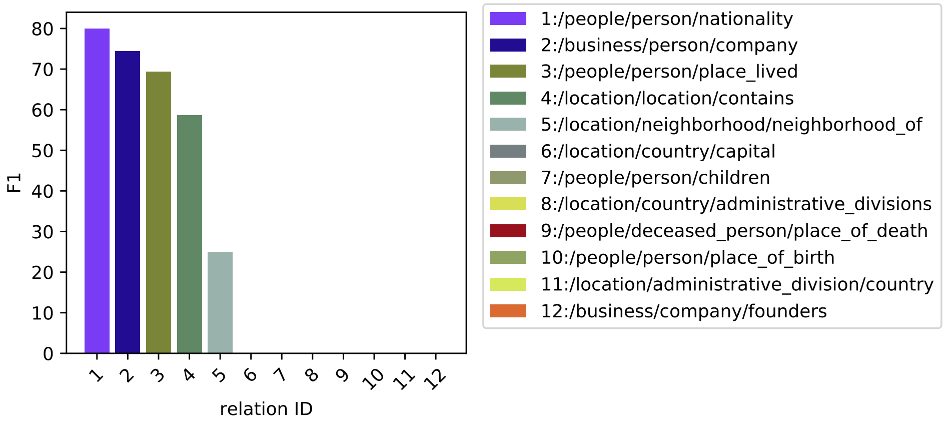
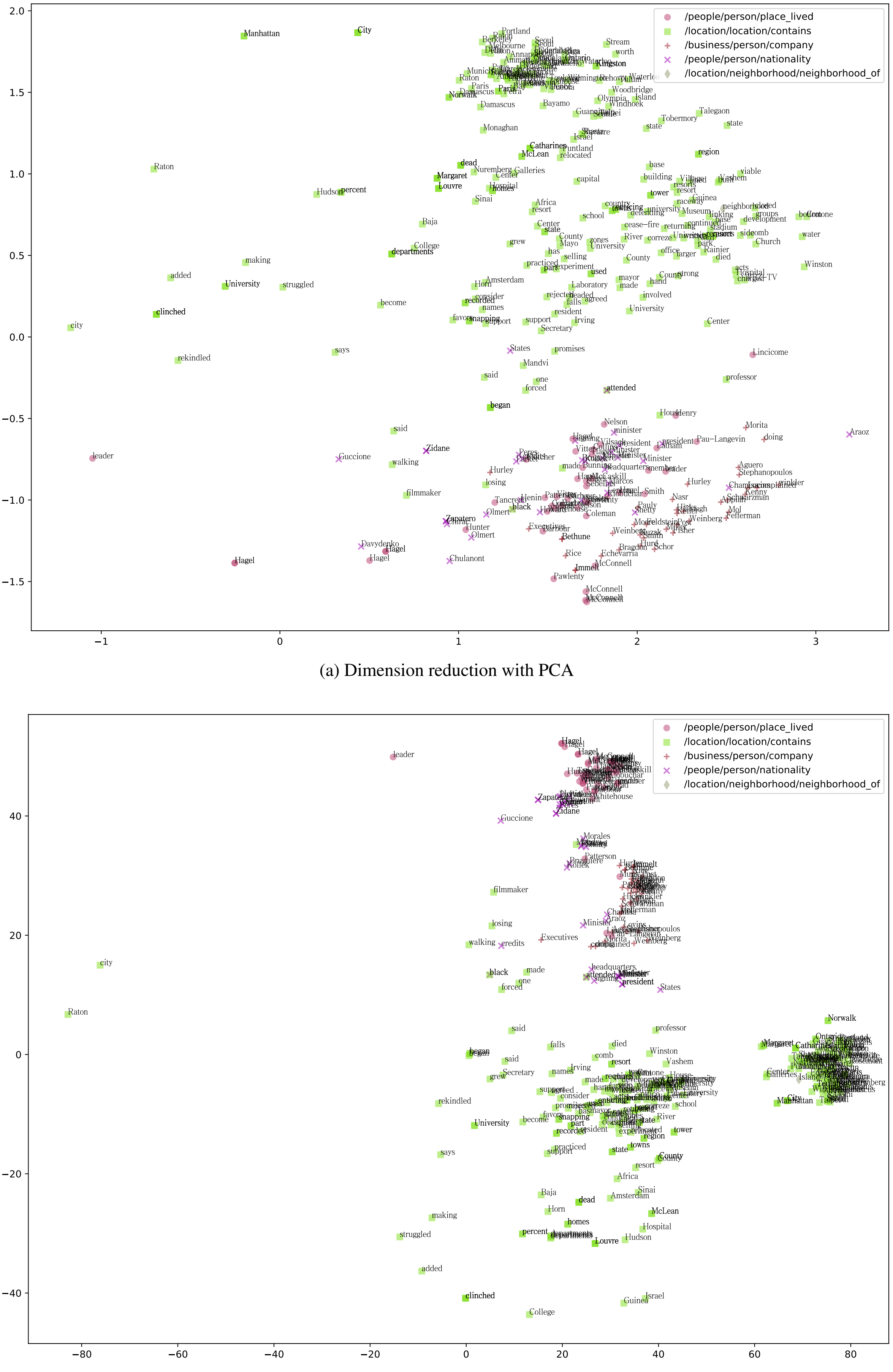


Figure 2

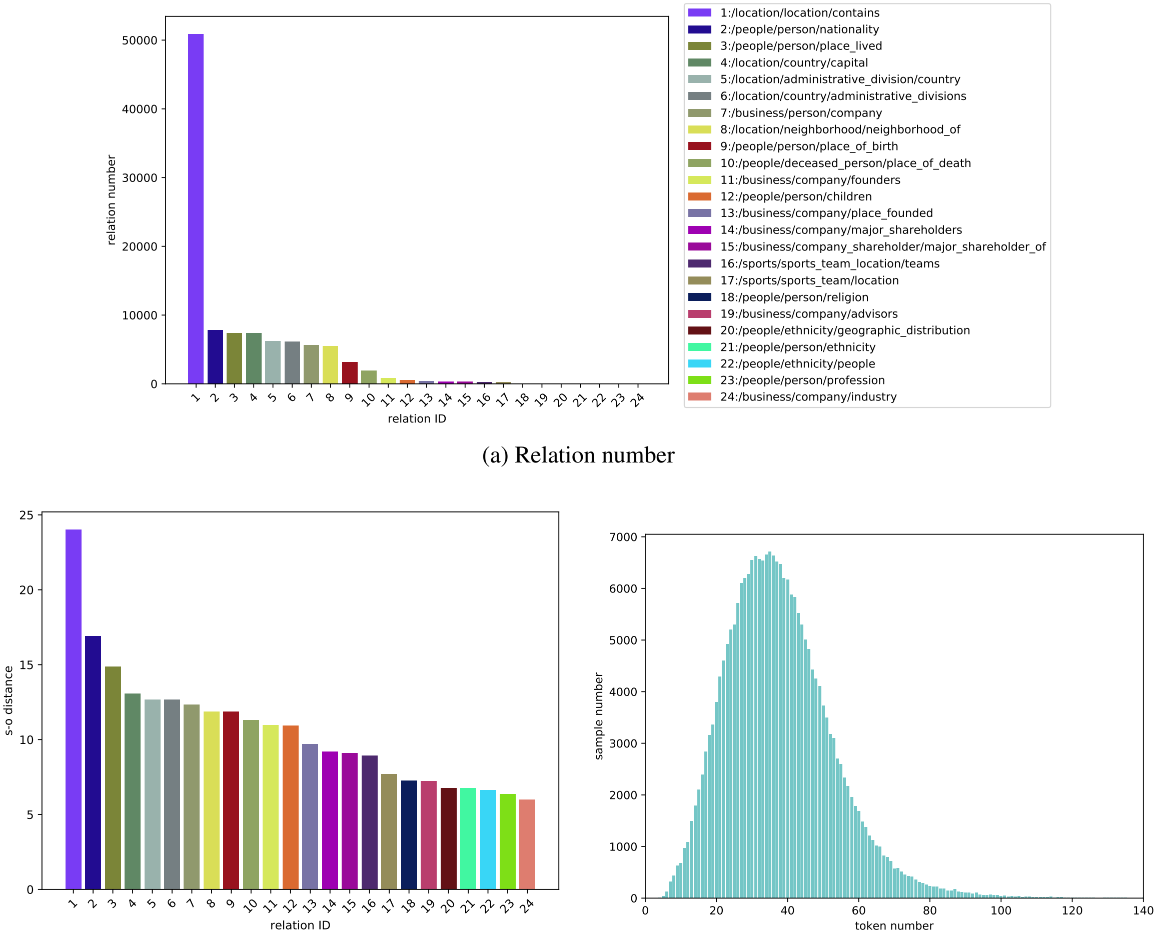
NYT dataset and use PCA and TSNE to make dimension reduction. As shown in Figure 3a, dominant vertices in GRNN for relation classification are represented as colored points placed according to their coordinates. Different colors denote different relation types. PCA tends to reveal the semantic relationship between relation types. the closely related relation types. We first observe that there are two main clusters, where the cluster of "*contains*" and "*neighbor_of*" relations describe the relationship between locations, while the other cluster describe the relation about person and organization/location. The lower part of the diagram shows the relations of "*place_lived*", "*company*" and "*nationality*" are closely related, but the classes are mixed. TSNE tends to show a more accurate relationship between relations. As shown in Figure 3c, the "*placed_lived*" is more close with "*nationality*". "*nationality*" is closely related to "*company*", while the "*place_lived*" is not always close to "*company*". This phenomenon is also in line with the actual situation. This GRNN, by leveraging vertex embeddings, provides better understanding about relations. This means the vertex embeddings encodes the semantic relationship between relation types.

## F Dataset Overview



(a) Dimension reduction with PCA

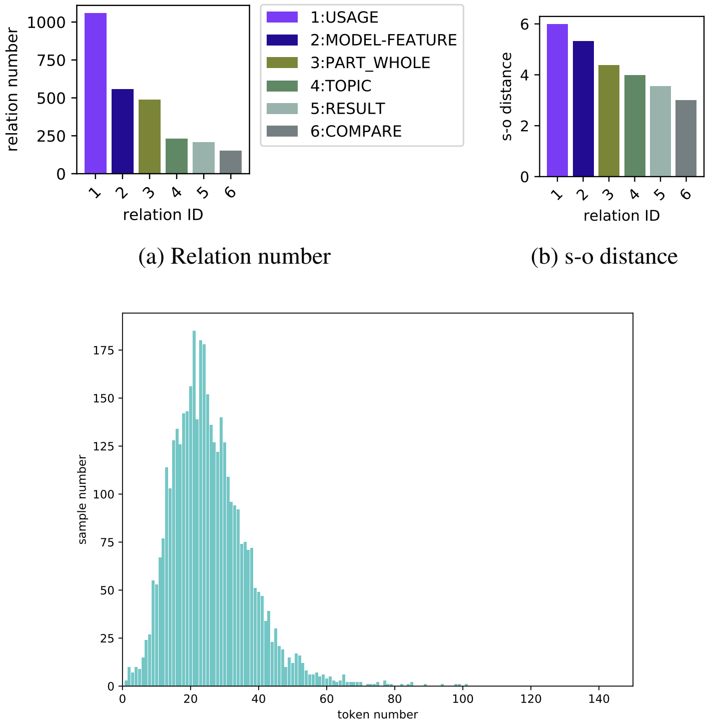
(b) Dimension reduction with TSNE



(a) Relation number

(b) s-o distance

(c) Token number



(a) Relation number

(b) s-o distance

(c) Token number