Learning Relatedness between Types with Prototypes for Relation Extraction

Lisheng Fu* **Ralph Grishman**
Computer Science Department
New York University
New York, NY 10003, USA
{lisheng, grishman}@cs.nyu.edu

Abstract

Relation schemas are often pre-defined for each relation dataset. Relation types can be related from different datasets and have overlapping semantics. We hypothesize we can combine these datasets according to the semantic relatedness between the relation types to overcome the problem of lack of training data. It is often easy to discover the connection between relation types based on relation names or annotation guides, but hard to measure the exact similarity and take advantage of the connection between the relation types from different datasets. We propose to use prototypical examples to represent each relation type and use these examples to augment related types from a different dataset. We obtain further improvement (ACE05) with this type augmentation over a strong baseline which uses multi-task learning between datasets to obtain better feature representation for relations. We make our implementation publicly available: https://github.com/fufrank5/relatedness

1 Introduction

Relation extraction identifies specific semantic relationships between two entities within a single sentence. For example, there is a Physical.Located relationship between *George Bush* and *France* in the sentence: *George Bush* traveled to *France* on Thursday for a summit. Relation extraction is a crucial task for many applications such as knowledge base population.

Relation schemas are mostly pre-defined in existing datasets. The definition of the relation type depends on the annotation guide. There is no clear intrinsic Ontology for relation types. In practice, relation types can be created based on interests. This leaves datasets with similar, related or overlapping schemas. For example, the annotation guides for Automatic Content Extraction (ACE) 03-05 changed from year to year. The later created Entities, Relations and Events (ERE) dataset was similar in the schema, but differs in details. Because of the difficulty of annotating relations, these datasets are all small individually and hard to be utilized together.

It is not an easy task to learn relatedness between relation schemas across different datasets since there is no instance-level labels available for the relatedness. However, we can observe the connections between the relation types from different datasets based on relation names or annotation guides. We propose to simplify the relatedness as binary (related or not) and to use manual review of relation names to decide the relatedness labels. This would give the prior knowledge that one relation type in one dataset may have closer relationships to some types than the others in another dataset. Then we design a model to recognize this similarity. We propose to use prototypical examples to represent each relation type. We rank these representations higher for related types, and lower for unrelated types using a pairwise loss function. Our base model is a multi-task learning model which focuses on learning a strong encoder using multiple datasets regardless of the relation schemas. We take the step further to explore utilizing the relatedness between the relation types. Experiments on ACE05 and ERE show that it can further boost the performance, especially in the low-resource settings.

2 Related Work

Relation type dependency: There have been a few ways to model the relationships between types in a multi-label relation dataset where we can learn

*This is the work that the author has done before joining Amazon Alexa AI.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 2011–2016
April 19 - 23, 2021.

the similarity or dependency from annotated examples. Surdeanu et al. (2012) used a two-layer hierarchical model. The object-level classifier is able to capture the label dependency, while the mention-level classifier is focused on multi-label classification. Riedel et al. (2013) used a neighborhood model to explicitly model the dependency between the labels in a matrix factorization framework. Both of models are designed to work on multi-label examples, which require annotation to capture the dependency between labels. In the recent work of neural methods for relation extraction, most of the work (Zeng et al., 2015; Lin et al., 2016; Liu et al., 2017) ignores the multi-label setting and does not explicitly model the label dependency. Ye et al. (2017), on the other hand, ranks the similarity between feature representation of the instance and the label embedding. In addition to ranking the positive classes higher than the negative ones, it ranks positive classes against each other to learn the connections between the positives classes. These methods all require annotated examples to learn the connections. In the case of relation types across different datasets, such annotation does not exist. We attempt to learn the similarity nevertheless using prototypes from each type.

Multi-task learning: Training multiple relation datasets at the same time could improve the robustness of the model and reduce annotation cost for relation extraction. (Fu et al., 2018) proposed to use a shared encoder to learn more general feature representation. We use a similar multi-task learning base model and incorporate the similarity between the relation schemas to further improve the performance.

3 Relation Model with Multi-task Learning

The majority of neural relation models (Zeng et al., 2014; Nguyen and Grishman, 2015b; Zeng et al., 2015; Lin et al., 2016) encode a sentence using a deep architecture to a vector representation followed by a softmax classifier, while the others (dos Santos et al., 2015; Ye et al., 2017) use a function to compute the score between label embedding and sentence representation. Inspired by Fu et al. (2018) where the shared encoder helps in the case of the multi-task learning, we choose the latter so that all relation types (including from different datasets) will share the whole model parameters except the label embeddings. Suppose we obtain the sentence representation $\phi(x)$ with a neural architecture. We define the label embedding as $W_l \in \mathbb{R}^D$, a D-dimension vector for each type. We compute the L_1 distance between them and learn a scoring function to estimate the scores $S_\theta(x)$ for every type:

$$S_\theta(x)_l = W_o \cdot |\phi(x) - W_l| + b_o, \quad (1)$$

where $W_o \in \mathbb{R}^D$ and $b_o \in \mathbb{R}$ are shared for all types.

We do not use the dot product (dos Santos et al., 2015; Ye et al., 2017) as the scoring function because the L_1 distance works slightly better in the multi-task learning experiments. The probability of every class is computed as the softmax output of the scores. Similar to (Fu et al., 2018), we jointly train two relation tasks at the same time with cross-entropy losses.

$$L = \lambda L_{r1} + (1 - \lambda) L_{r2}, \quad (2)$$

where L_{r1} and L_{r2} are the cross-entropy losses for the two relation tasks. λ is the hyperparameter to control the learning speed between the two tasks. This would give a strong baseline of utilizing the two datasets together.

3.1 Prototypes of Relation Types for Learning Similarity

For each relation type, we randomly select k examples (S_k) from the training set as supporting examples. We use the mean of the representations of these examples as the prototype for the relation type:

$$\bar{x}_c = \frac{1}{k} \sum_{x_i \in S_k} \phi(x_i) \quad (3)$$

These prototypes are inspired by the Prototypical Networks (Snell et al., 2017). However, in the training procedure, these supporting examples are randomly selected for every mini-batch. We have dynamic prototypes during training.

We define $S_\theta(\bar{x}_c)_l$ as the similarity score to type c for type l. We hypothesize that if the two relation types are similar in semantics, they should obtain high similarity score. Within the dataset, if the relation types in the schema are mutually exclusive to each other, then we would expect a high similarity score to itself and low scores to the other types. Across the datasets, the prototypes would obtain high scores for related types and low scores in the

unrelated types. We use a pair-wise ranking loss (dos Santos et al., 2015) to learn this relatedness across the datasets.

For $l \in L$ and $c \in C$, $S_\theta(\bar{x}_c)_l$ gives the score for the similarity between the type l in the relation schema L and the type c in the relation schema C. Let $c^+ \in C$ be a class related to l and $c^- \in C$ be a class unrelated to l. The similarity scores would be $S_\theta(\bar{x}_{c^+})_l$ and $S_\theta(\bar{x}_{c^-})_l$ respectively. We define the pair-wise ranking loss as:

$$\begin{aligned} L_s = & log(1 + exp(\gamma(m^+ - S_\theta(\bar{x}_{c^+})_l)) \\ & + log(1 + exp(\gamma(m^- + S_\theta(\bar{x}_{c^-})_l)) \end{aligned} \quad (4)$$

m^+ and m^- are the margins and γ is the scaling factor. This loss function would push $S_\theta(\bar{x}_{c^+})_l$ higher for related type pair between c^+ and l and $S_\theta(\bar{x}_{c^-})_l$ lower for unrelated type pair between c^- and l. We manually create a relatedness matrix to state whether the two types are related or not between the types in C and L based on the definition of the relation types. For each step of training, we pick the highest scored c^- from unrelated types and lowest scored c^+ for related types corresponding to type l.

$$c^- = \underset{c \in C^-}{argmax} S_\theta(\bar{x}_c)_l \quad (5)$$

$$c^+ = \underset{c \in C^+}{argmin} S_\theta(\bar{x}_c)_l \quad (6)$$

where C^- are types unrelated to l and C^+ are types related to l. In experiments, we use looser margins ($m^+ = 0.5$, $m^- = 0.5$, $\gamma = 1.0$) compared to (dos Santos et al., 2015) as we are learning the relatedness between types rather than doing classification for individual instance. The ranking loss is jointly trained as an auxiliary task with the main relation tasks:

$$L = \lambda L_{r1} + (1 - \lambda) L_{r2} + \beta L_s, \quad (7)$$

where we use β to control the weight for learning the relatedness. If β is too large, it could disrupt the learning of main relation tasks. With appropriate weight, it could help augment the label embeddings for the relation types by considering the similarity between them.

4 Experiments

4.1 Datasets

We select two datasets with similar relation schemas in this experiment. There is overlapping of relation types between ACE05 and ERE, but the annotation guides are different in details (Aguilar et al., 2014). Thus, we can not combine the training data directly as the same type. Doing so would actually lead to worse result as it introduces more noise than benefit. The multi-task learning would be a better choice at this setting. We take a step further and try to learn the similarity between the types at the same time. There are 6 main semantic types in ACE and 5 in ERE. Manual review of the relatedness (related or not) is trivial in this case because the relation names are almost identical for related relation types. In practice, it may take a few minutes to review more complicated relation schemas, but it would cost significantly less than annotating on the instance-level in text. For preprocessing the data, we follow previous work (Gormley et al., 2015; Nguyen and Grishman, 2015a; Fu et al., 2017, 2018) on ACE05. It contains 6 domains: broadcast conversation (bc), broadcast news (bn), telephone conversation (cts), newswire (nw), usenet (un) and weblogs (wl). We use newswire as training set (bn & nw), half of bc as the development set, and the other half of bc, cts and wl as the test sets. We followed their split of documents and their split of the relation types for asymmetric relations (directionality taken into account expect for *physcial* and *person-social* types). We perform the same preprocessing for the ERE dataset, which contains documents from newswire and discussion forums. We follow the document split from (Fu et al., 2018).

4.2 Multi-task Learning Baseline

Following previous work (Nguyen and Grishman, 2015a; Fu et al., 2018), We use a similar encoder to obtain the feature representation $\phi(x)$ as our baseline. The input layer is the concatenation of word embedding, entity embedding and position embeddings. We use pretrained word2vec (Mikolov et al., 2013) as the word embedding with embedding size d_w. The entity embedding and position embeddings are randomly initialized vectors according to the entity type of the token and relative distance to the two arguments of the relation. The embedding sizes are d_e and d_p respectively. We follow previous work for these input embedding sizes as $d_w, d_e, d_p = 300, 50, 50$. It is followed by Bidirectional RNN with attention and a fully connected layer to match the size for the label embedding. We use 150 for the RNN state size

	ACE05			
Method	**bc**	**wl**	**cts**	**avg**
Single-task	60.22	53.77	52.01	55.33
Multi-task	60.60	**56.20**	56.72	57.84
+ Relatedness	62.05	56.10	**59.12**	**59.08**
(Fu et al., 2018)	61.67	55.03	56.47	57.72
+ Regularization	**62.24**	55.30	56.27	57.94

Table 1: Learning the relatedness between types (full training set).

	ACE05			
Method	**bc**	**wl**	**cts**	**avg**
Single-task	54.80	47.27	48.42	50.17
Multi-task	56.67	51.39	55.23	54.43
+ Relatedness	**58.31**	**53.13**	**56.50**	**55.98**
(Fu et al., 2018)	57.39	51.44	54.28	54.37
+ Regularization	57.73	52.30	54.63	54.89

Table 2: Learning the relatedness between types (50% training).

and 200 for the label embedding size. The output of this encoder is $\phi(x)$. Then we can perform classification using the scores obtained from Equation 1.

In a mini-batch of training step, we randomly select examples from both datasets proportionally according to the dataset size so that the model can finish reading both datasets at the same time after every epoch. Because the difference of the number of examples for the two datasets in the batch, we set $\lambda = \lambda_d \frac{|D_1|}{|D_1|+|D_2|}$, where $|D_1|$ and $|D_2|$ are the number of examples for each dataset in a single batch. In a special case where the two datasets are actually split from one original dataset, we can set $\lambda_d = 1.0$, and then the two datasets are going to be learned at the same speed. In our case, we use $\lambda = 0.8$ so that the two relation tasks are roughly learning at the same speed. As the result, our multi-task model using label embedding is comparable to (Fu et al., 2018) (Table 1), which serves as a strong baseline since it is already better than training a single relation task. We obtain all our scores as the average of 10 runs to report stable results.

4.3 Learning the Relatedness between Two Relation Schemas

By learning the relatedness at the same time (Equation 4,7, $\beta = 0.001$), we obtain better results at the full training set (Table 1). The improvement is more obvious with a smaller training set (Table 2 at 50%). The regularization in the previous work does not take the relatedness on the type-specific basis into account, which fails to obtain clear improvement over the multi-task baseline. Our method is more effective in incorporating additional knowledge from multiple sources. We also set up a low-resource setting where we only have N examples for each relation type (Figure 1 at $N = [10, 20, 30, 40, 50]$). The negatives are randomly selected according to the pos/neg ratio. We can observe larger improvement with less training data. This is also to consider the skewed data distribution in the dataset where there are far more examples for some types than the others. The k supporting prototype examples are drawn randomly at every step. We use $k = 50$ for the experiments and $k = N$ for the low-resource settings. Overall, the improvement is impressive, especially for the low-resource settings. It is also worth to note that the single task models for these low-resource settings obtain virtually zero scores without multi-task learning as there is not enough data to train the encoder. The multi-task learning between two relation tasks is better than training on a single task and more effective for a smaller training set. We now show that learning the relatedness between the types could further improve the model.

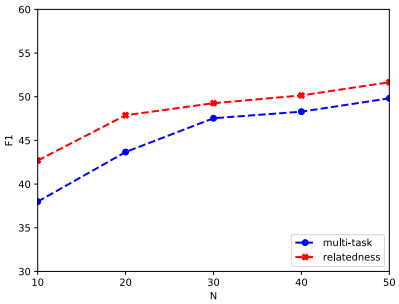

Figure 1: Low-resource setting with N examples for each positive relation type on ACE05.

5 Conclusion

We use prototypes of relation types to learn the relatedness between them in a multi-task learning framework. With prior knowledge of relatedness between relation types, the model obtains further improvement in addition to sharing the encoder of the sentence. The prior knowledge is ob-

tained through manual review of relation names, which costs significantly less than annotating on the instance-level in text. In this paper, we simplify the relatedness as binary. It would be interesting to further explore the relationships between relation types as a more dynamic metric.

References

Jacqueline Aguilar, Charley Beller, Paul McNamee, Benjamin Van Durme, Stephanie Strassel, Zhiyi Song, and Joe Ellis. 2014. A comparison of the events and relations across ace, ere, tac-kbp, and framenet annotation standards. In *Proceedings of the Second Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 45–53.

Lisheng Fu, Bonan Min, Thien Huu Nguyen, and Ralph Grishman. 2018. A case study on learning a unified encoder of relations. In *Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text*, pages 202–207, Brussels, Belgium. Association for Computational Linguistics.

Lisheng Fu, Thien Huu Nguyen, Bonan Min, and Ralph Grishman. 2017. Domain adaptation for relation extraction with domain adversarial neural network. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 425–429, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Matthew R. Gormley, Mo Yu, and Mark Dredze. 2015. Improved relation extraction with feature-rich compositional embedding models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1774–1784, Lisbon, Portugal. Association for Computational Linguistics.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2124–2133, Berlin, Germany. Association for Computational Linguistics.

Tianyu Liu, Kexiang Wang, Baobao Chang, and Zhifang Sui. 2017. A soft-label method for noise-tolerant distantly supervised relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1790–1795, Copenhagen, Denmark. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Thien Huu Nguyen and Ralph Grishman. 2015a. Combining neural networks and log-linear models to improve relation extraction. *arXiv preprint arXiv:1511.05926*.

Thien Huu Nguyen and Ralph Grishman. 2015b. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 39–48, Denver, Colorado. Association for Computational Linguistics.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 74–84, Atlanta, Georgia. Association for Computational Linguistics.

Cícero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 626–634, Beijing, China. Association for Computational Linguistics.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465, Jeju Island, Korea. Association for Computational Linguistics.

Hai Ye, Wenhan Chao, Zhunchen Luo, and Zhoujun Li. 2017. Jointly extracting relations with class ties via effective deep ranking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1810–1820, Vancouver, Canada. Association for Computational Linguistics.

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.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*,

pages 2335–2344, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.