Harvesting Mature Relation Extraction Models from Limited Seed Knowledge: A Self-Development Framework for DS Rule Expansion

Anonymous ACL submission

Abstract

Distantly-supervised relation extraction (DSRE) is an effective method to scale relation extraction (RE) to large unlabeled corpora with the utilization of knowledge bases (KBs), but suffers from the scale of KBs and the introduced noise.

To alleviate the above two problems, we propose a novel framework called **S**elf-devel**O**pment r**U**le ex**P**ansion (**SOUP**), which starts from limited amount of labeled data and continuously produces low-noise labels on large-scaled unlabeled data by a growing learnable logical rules set.

Specifically, SOUP achieves a mutual enhancement of RE model and logical rules set, first a RE model is trained on the labeled data to summarize the knowledge, then the knowledge is utilized to explore candidate rules from unlabeled data, finally high-quality candidates are selected in a graph-based ranking manner to extend the logical rules set and new rule-labeled data are provided for better RE model training.

Experiments on wiki20 dataset demonstrate that, with limited seed knowledge from small-scaled manually labeled data, SOUP achieves significant improvement compared to baselines by producing continuous growth of both logical rules and the RE model, and that labeling noise of SOUP is much less than DS. Furthermore, RE model enhanced by SOUP with 1.6k logical rules learned from prior knowledge could produce an equivalent performance to the model trained on data labeled in DS manner by 72k relational facts of KBs.

1 Introduction

Relation extraction (RE) is a fundamental task in NLP, which aims at inferring the relation r of entity pairs (e_1, e_2) from plain texts. Since sufficient manually annotated training data is expensive, distant supervision (DS) is commonly adopted to collect abundant weakly-labeled data by applying the following labeling heuristic on knowledge bases: for any relational fact (e_1, e_2, r) in a knowledge base, all the sentences mentioning both entities e_1 and e_2 may express relation r and thus are assigned with the weak label r.

However, in real-world scenarios, the feasibility and performance of DSRE may be limited: (1) the large-scale and high-quality KBs might be incomplete or even unreachable, especially for emerging industries where new relation types of interest are being explored. (2) The labeling heuristic of DS intrinsically introduces noise into the weak labels by solely considering the entities' co-occurrence. As illustrated in the follow examples, the expressed relation type of the same entity pair may vary based on the different context information, such as trigger words (sentences S_1 and S_2) or dependency information (sentence S_3).

S_1: **Trump** lives in **United States** with family. => ***live_in***
S_2: **Trump** was born in **United States**. => ***born_in***
S_3: In **United States**, teachers from rhode island met the 45^{th} president, **Trump**. => ***president_of***

With the awareness of the second deficiency of DSRE, lots of prior work has been done to denoise the weakly-labeled dataset or to design robust RE systems against the noise (Lin et al., 2016; Christopoulou et al., 2021). However, less exploration has been investigated to seek the remedy of the first problem for DS. (Qu et al., 2018).

To alleviate the above two problems simultaneously, we propose a novel self-development rules expansion framework (SOUP). As shown in Figure 1, SOUP starts from seed knowledge in small-scaled labeled data, and continuously explores new low-noise weakly-labeled data by mutually enhancing the growable logical rules and RE model iteratively through following 3 procedures: (1) **From Data to Rules:** given the pseudo-labeled data extracted from RE model's high-confidence predictions on unlabeled dataset, a Rule Generator is

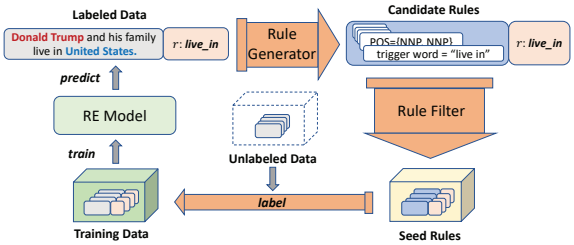

Figure 1: An illustration of how SOUP explores new knowledge from unlabeled data.

applied to extract its contextual feature and transfer it to a logical rule with learnable parameters, which are taken as a candidate rule. (2) **From Rules to Rules:** with the prior that rules of the same relation type r are supposed to be semantically similar, we further propose a Rule Filter to regularize rules' representations and divide candidate rules into positive high-quality ones and negative noisy ones, based on their semantic distance with seed rules. Then the seed rules set is enriched by positive candidates. (3) **From Rules to Data:** Since the logical rule considers various context information (including entity information and so on), it can be easily generalized to unlabeled data with unseen entity pairs and provide low-noise labeled data to enrich RE training set. SOUP is more desirable in real-world scenarios especially in emerging domains, as it can grow rapidly and brings better performance with fewer initial seed resources.

We evaluate SOUP on a public RE dataset wiki20 (Gao et al., 2021) to show its improvements towards baselines. With the same RE model structure, DS requires data labeled by about 72k relational facts from KB to achieve the Macro F1 score of 60.08, while SOUP could achieve the identical performance with only 1.6k initial seed rules extracted from 30k facts, which demonstrates SOUP is more resource-economical, relying on its great knowledge summarizing and learning ability. We further conduct experiments and ablation studies with in-depth analysis to verify that SOUP brings much less labeling noise and collaborates different modules effectively.

Our contributions are threefold:

1. We propose a logical rule generation method for RE, which has learnable parameters and considers various contextual features to provide low-noise relation labels.

2. We design a novel self-development rule expansion framework SOUP, which starts from a few manually labeled data and continuously improves the logical rules and RE model based on large-scale unlabeled data.

3. Experiments on the public dataset verify that SOUP can grow up rapidly with low-noise rule-labeled data and less requirement for initial seed knowledge.

2 Related Work

Distant Supervision Since RE models' performance is commonly limited by the scale of human-annotated data, distant supervision (DS) is introduced to collect the large-scale training data for RE models in an economical way by aligning entity pairs in sentences to relational facts in existing knowledge bases. However, the context-agnostic labeling heuristic of DS inevitably brings the labeling noise and thus hurts the model's performance. To reduce the training noise for RE models, many DS-based methods (Hoffmann et al., 2011; Surdeanu et al., 2012) treat DSRE as a multi-instance learning task by bagging sentences based on entity pairs and predicting involved relation types at bag level. To select the most informative instance inside a bag for training, (Lin et al., 2016) proposes an intra-bag attention mechanism distributing weights to the sentences within the bag. However, KBs are not always available and usually expensive to construct, especially for emerging industries.

Weakly Supervision To reduce the labeling dependency for KBs, some work has explored constructing logical rules automatically from limited labeled seed data and large-scale unlabeled data. (Batista et al., 2015) proposes to iteratively generate rules from centroids of the seed data and enlarge the seed data by labeling unlabeled data with rules. Since new rules are directly generated from seeds labeled by old ones, the noise introduced in the seeds by rules can be accumulated. (Qu et al., 2018; Gao et al., 2020) try to cooperate rules and RE model, which utilize rules to seek new seed data from unlabeled data and learn a RE model based on the updated seed dataset to explore more reliable rules. However, the independent rule learning manner for different relations may make them weak in distinguishing semantically similar relation types (for instances, *subsidiary* and *member of*). To provide low-noise rules, SOUP models semantic features for different relation rules and selects high-quality ones by

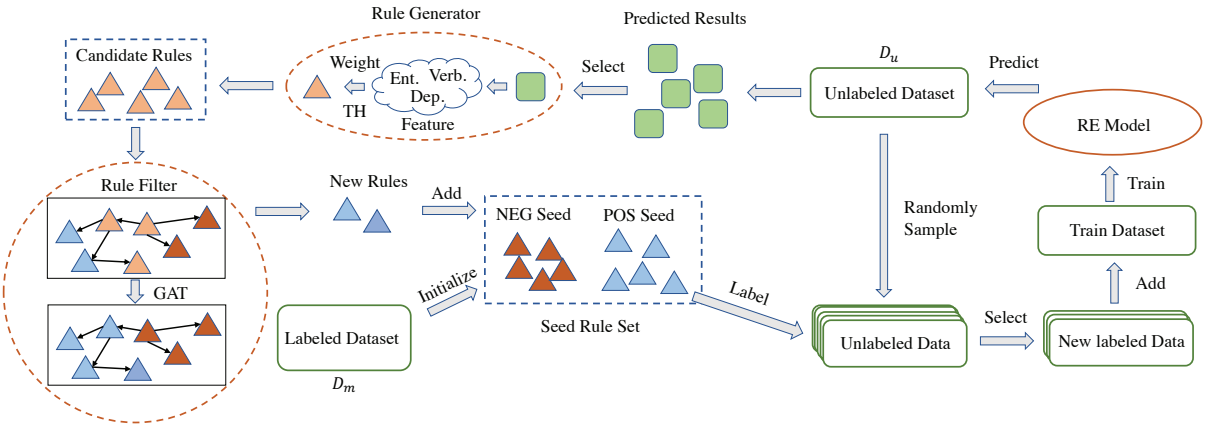

Figure 2: The framework of SOUP. The initial seed rule set is extracted from D_m, then a train dataset is initialized by rule-labeled data and applied to learn a RE model. After that, SOUP iterates among three procedures: (1) generate candidate rules from model's prediction on D_u; (2) filter reliable rules from candidates to extend the seed rule set; (3) add new rule-annotated data to enrich the train dataset and power the RE model.

ranking their relative distance in a graph manner.

3 Methodology

Overview In this section, we introduce SOUP in detail, which starts from a small-scale manually labeled dataset D_m and achieves mutual enhancement between logical rules and the RE model on a large-scale unlabeled dataset D_u. As is shown in Figure 2, SOUP consists of three components: (1) a *Rule Generator* to summarize contextual features contained in high-confidence model-labeled instances from D_u, and build candidate logical rules with learnable parameters; (2) a *Rule Filter* to enrich the seed rule set by filtering the candidate rules with a semantic graph of rules; (3) a *RE Model* trained on the labeled dataset to explore knowledge from D_u for new candidate rules. Notice that the seed rule set is initialized by applying Rule Generator to build logical rules from D_m and a scoring function for further filtering (explained in 3.2.1).

3.1 Rule Generator: From Data to Rules

Existing methods fail to take into account the contextual features of labeled data to build logical rules, rendering the limited ability to generalize and introducing much noise into the labeling process. In this section, we formulate our rule representations based on contextualized features, which allows for a more robust knowledge summarization of labeled data and better transfer to unlabeled samples.

Rule Definition Our logical rule $isObey_R(\cdot)$ is a function with bool output, which contains a set of concerned features $R_f = \{EF, TWF, DSF\}$ (*i.e.*, entity feature, trigger words feature and dependency structure feature), corresponding importance weight w_i and similarity measurement $Sim_i(\cdot,\cdot)$ ($i \in \{1, ..., |R_f|\}$), a threshold TH, and a target labeling relation r.

$$isObey_R(x) = \sum_{i=1}^{|R_f|} Sim_i(R_{f_i}, x) w_i \geq TH \quad (1)$$

R_{f_i} is the i^{th} concerned features of R_f ($|R_f| = 3$). For a unlabeled data x, it matches this rule and can be labeled with r if $isObey_R(x)$ is `TRUE`.

Rule Generator Process The Rule Generator transfers a labeled instance to a logical rule automatically by: (1) collecting the concerned features R_f of the labeled data; (2) computing the learnable parameters $\{w_i\}$ and TH. We elaborate the two steps in following subsections.

3.1.1 Feature Collection

Considering that instances with similar contextual features are more likely to express the same relation, we distinguish different relations from the following three context-aware features: (1) entity features, (2) trigger word features and (3) dependency structure features.

Entity Features (EF) Our entity features include two aspects: (1) the POS feature pos_i for each entity mention e_i from Stanford CoreNLP toolkit (Manning et al., 2014) (2) the entity's concept embedding e_i^c by retrieving top 10 possible concepts of e_i with Microsoft Concept Graph (Wu et al., 2012) and take the average of their embeddings (Shalaby et al., 2019).

Trigger Word Features (TWF) Given the trigger word v whose involved entity pair is ep=$\{e_1$,$e_2\}$, our trigger word features include: (1) semantic roles srl of ep from PropBank (Bonial et al., 2014), (2) thematic roles trl of ep and event information $\mathcal{E}$ using VerbNet (Brown et al., 2019) (Our work is the first to utilize this toolkit in relation extraction), and (3) the trigger embedding e^v by extracting the verb phrase with OpenIE toolbox (Christensen et al., 2011) and computing its embedding by BERT (Devlin et al., 2019).

Dependency Structure Features (DSF) We collect the shortest path $\mathcal{P}$ information (containing both nodes and edges) between the entities in the dependence structure.

Similarity Measurement Given a unlabeled instance x and a logical rule, we compute their similarity on each type of contextual feature R_{f_i}, donated as $Sim_i(R_{f_i}, x)$, which is used in Eq. 1 to help infer if x belongs to the target relation of the rule. (*Please refer to Appendix A.1 for more details on the corresponding similarity measurements*).

3.1.2 Weight Distributor

To comprehend the relation expressed in a sentence, human usually consider various contextual information and focus on those crucial for relation judgment. Based on this motivation, we assign weights $\{w_i\}$ to our contextual features to distinguish their contributions to rules. The weight assignment is independent for different rules.

If a type of feature is crucial for judgement, it must work better in filtering sentences expressed the target relation than any other features. Thus, for each rule with target relation r, to model its contextual features' relevance to r, we warp each R_{f_i} in a single-feature rule $isObey_{R_{f_i}}(\cdot)$. Then we estimate R_{f_i}'s importance by collecting all the instances matching $isObey_{R_{f_i}}(\cdot)$ and computing their semantic relevance to r. $isObey_{R_{f_i}}(\cdot)$ is definited as follows:

$$isObey_{R_{f_i}}(x) = Sim_i(R_{f_i}, x) \geq TH_i \qquad (2)$$

where TH_i is manually set by workers.

Rule's embedding For further computation, we obtain the rule's embedding as the average embedding of all the instances in D_m that make the rule's output `TRUE`. The instance's embedding is produced by the BERT encoder (Han et al., 2019) trained on D_m in a supervised manner.

Ranking Model Motivated by the translation quality estimation work (Zhang and van Genabith, 2020), which ranks the translation by their semantic distance towards the source sentence. Since it is difficult to compute an exact value for feature's importance, we try to measure the relative importance among the three types of feature towards the target relation r by ranking the three single-feature rules based on their semantic relevance to r.

To measure the relevance, a Ranking Model $RM(Q_i, K; W)$ (parameterized by W) is trained to compute the attention of rules $\{isObey_{R_{f_i}}\}$ ($i \in \{1, 2, 3\}$) towards relation r based on their semantic representation:

$$RM(Q_i, K; W) = Attn(Q_i, K; W) = Q_i^{\mathsf{T}} W K$$

Q_i and K are the embedding of $isObey_{R_{f_i}}$ and relation r. The relation r's embedding is obtained from pre-trained BERT-Medium. Finally, we sort $\{s_i = RM(Q_i, K; W)\}$ ($i \in \{1, 2, 3\}$) decreasingly to get $isObey_{R_{f_i}}$'s rank result p_i.

Weight Mapping Based on a rank-to-weight list $r2w = \{w_j\}_{j=1}^{3}$ set by workers, weight w_j is assigned to the R_f element who ranks in j^{th} position.

Objective We apply the metric-driven loss function LambdaLoss (Wang et al., 2018) as the objective to optimize RM, which takes both scores and ranks into consideration and encourages the single-rule more related to r to get a higher attention score:

$$l(s, y) = -\sum_{y_i > y_j} log_2 \sum\nolimits_{\pi} \left(\frac{1}{1 + e^{-\sigma(s_i - s_j)}}\right)^{(\rho_{ij} + \mu\delta_{ij})|G_i - G_j|} \qquad (3)$$

where s_i and y_i are the predicted attention score and gold rank of i^{th} single-feature rule. Other variables' definition is same as (Wang et al., 2018).

Training data We transfer each labeled data in D_m to a logical rule $isObey_R$ and generate three single-feature rules $\{isObey_{R_{f_i}}\}$ ($i \in \{1, 2, 3\}$) for it. To get the gold rank y_i for i^{th} single-feature rule, we evaluate its performance on all the data in D_m. Since the rule with higher labeling precision should get higher evaluation score, scoring function is:

$$Sc = \frac{F}{N} log_2(F) \qquad (4)$$

where F is the number of instances correctly labeled by the rule, N is the number of instances with `TRUE` rule output. Thus, we regard $\{y_i\}$ as the supervision signal by sorting $\{isObey_{R_{f_i}}\}$ ($i \in \{1, 2, 3\}$) decreasingly with respect to their Sc.

3.1.3 Threshold Selector

For some rule, each of its contextual features can lead to a correct judgment for the target relation, while for others it's more reasonable to consider several features together for reliable labeling, thus different rules should have different thresholds.

We first set a candidate threshold set THS and for each rule, a suitable threshold is selected from THS. Similar as 3.1.2, we warp each $TH_p \in THS$ in a can-threshold rule defined as:

$$isObey_{R_p}(x) = \sum_{i=1}^{|R_f|} Sim_i(R_{f_i}, x) w_i \geq TH_p \tag{5}$$

Then, by applying the Ranking Model (trained in Weight Distributor) on $\{isObey_{R_p}\}_{p=1}^{|THS|}$, The candidate threshold with top rank can-threshold rule is selected as TH for $isObey_R$.

3.2 Rule Filter: From Rules to Rules

This module aims to filter out noisy candidate rules and enrich seed rule set with high-quality ones based on their semantic relevance. We propose to formulate it as a semi-supervised nodes classification task on the semantic graphs of the rules with a Rule Classifier. We first introduce how to select rules, then elaborate the Rule Classifier.

3.2.1 Metrics for Selecting Rules

Seed Rules To initialize the seed rule set, we first apply the Rule Generator to transfer every instance in D_m to a rule, then rank the rules with respect to their performance score on D_m data (calculated as Eq. 4). Finally for each relation, the top N_I rules with the highest scores are chosen as initial positive seeds while the randomly sampled N_I positive seeds from all other relations are taken as initial negative seeds. In each iteration, The Rule Classifier is applied to pick new positive seeds from candidates and update negative seeds.

Candidate Rules In each iteration, we apply the RE model to D_u and utilize Rule Generator to transfer the top N_C predictions with highest confidence to rules, which are taken as candidates.

3.2.2 Rules Classifier

For each relation, this module picks new seed rules in four steps: (1) Construct a graph of both seeds and candidates belonging to this relation; (2) Initialize nodes' representation by rules' embedding (introduced in 3.1.2); (3) Update nodes' representation by feature propagation; and (4) Pick N_S candidates as new positive seeds based on a marginal-based ranking method.

Graph Construction Since semantically similar rules are more likely to express the same relation, we build a k-nearest neighbor graph $G_r = \{V = V_c \bigcup V_s, A\}$ for relation r to model the semantic relevance among rules, where V_c, V_s are nodes for candidate and seed rules, A is the adjacency matrix.

Rule Feature Propagation For relation r and its corresponding graph G_r, we apply an independent 2-layer graph attention network (Veličković et al., 2018) to propagate nodes' semantic information. Given node i's representation h_i and its neighbors N_i, in each layer the attention weight between node i and j can be computed as:

$$a_{ij} = \frac{exp(f(A^\mathsf{T}[Wh_i, Wh_j]))}{\sum_{k \in N_i} exp(f(A^\mathsf{T}[Wh_i, Wh_k]))} \tag{6}$$

where W is parameter matrix and f is LeakyReLu function. Then the node representation of the next layer is updated as:

$$h_i^* = a_{ii} W h_i + \sum\nolimits_{j \in N_i} a_{ij} W h_j \tag{7}$$

Objective Rule Classifier's objective is defined as:

$$L = L_{sup} + L_{nei} \tag{8}$$

where $L_{sup} = -\sum (y_i log p_i + (1 - y_i) log(1 - p_i))$ $(i \in V_s)$ is the supervised loss, y_i is seed nodes i's binary label, p_i is the predicted possibility of node i being POS, and $L_{nei} = \sum_{j \in V} \sum_{k \in N_j} \|h_j - h_k\|_2$ is to encourage neighbor nodes to be close to each other in semantic space.

Seed Rules Extension To filter the most semantically distinguishable candidates, with the trained Rules Classifier for relation r, we (1) Compute the centroids' representation of positive and negative seeds as the average of corresponding nodes' representation; (2) Compute the cosine similarity of each candidate to the two centroids as S_{pos} and S_{neg}, (3) Rank all candidates by $S_d = S_{pos} - S_{neg}$ and pick the top N_S as new positive seeds.

3.3 RE Model: From Rules to Data

The BERT-AVG introduced in Gao et al. (2021) is taken as our RE model, which consists of a BERT-based sentence encoder and a classifier. By feeding the average of bag-level sentence representation to classifier for prediction, this network can alleviate the noise influence in current knowledge and make better candidate rules' label prediction.

Training Data In each iteration, we randomly sample N_u unlabeled instances from D_u and filter high-quality rule-label results as follows: we first label each unlabeled data by the matched seed rule whose embedding has the highest cosine similarity Sc_{sim} with its, then rank all the labeled data by Sc_{sim} and choose the top N_t ones to enrich the training set of RE model.

4 Experiments and Analysis

4.1 Dataset and Evaluation Metrics

We conduct our experiments on wiki20 dataset, proposed in (Han et al., 2020) by aligning Wikipedia corpus with Wikidata (Vrandečić and Krötzsch, 2014) facts (triplets). We adopt the re-organized wiki20 dataset (Gao et al., 2021), which contains 80 actual relation types and a type *NA* (the relation is not in wiki20's relation ontology). Since *NA* covers many unseen relations and it's unreasonable to distinguish candidate rules of *NA* into POS or NEG, we only focus on the other 80 relation types.

By discarding the distantly supervised label, we take wiki20 training set as D_u. 30k human-labeled instances are evenly sampled from wiki80 dataset (Han et al., 2019) and taken as D_m. We evaluate the RE model performance on wiki20's test set with following metrics: Area Under Curve (AUC) values, Micro F1, Macro F1 and Precision@N (P@N). Our dataset's statistics is shown in Table 1. Notice that we exclude all the test instances included in D_m to make sure that there is no overlaps between our test set and D_m. (*Please refer to Appendix A.2 for details on parameters setting.*)

Dataset	# Facts	# Sentences	# Instances
D_u	103713	154339	295594
D_m	30077	29390	30080
RE model Validation	17484	15356	17484
RE model Test	28150	54118	56686

Table 1: The statistics of the dataset in our experiments.

4.2 Baselines

Our baselines contains weakly supervised frameworks and distant supervised frameworks.

REPEL (Qu et al., 2018) learns a triplets evaluation function on rule-labeled data and picks new rules (shortest dependency path) generating high-quality triplets iteratively.

Snowball (BERT) (Gao et al., 2020) first trains a Relational Siamese Networks (RSN) and a Relation Classifier on prior triplets. Then for a new relation, Snowball extracts rules (entities co-occurrence) from the given seed data for this relation, and filter new seed data by rules and the above two modules, which fine-tuned by the seed data.

DSRE BERT-AVG (Gao et al., 2021) The same RE model structure as SOUP but trained on the dataset built by triplet-labeled data based on the prior knowledge in KB.

DSRE VAE (Christopoulou et al., 2021) is a distant supervised framework. Beside the triplet-labeled training data, this method also applies pre-trained entity link prediction model TransE (Bordes et al., 2013) to improve sentence expressivity by sentence reconstruction.

For **REPEL**, to follow its original setting, we arrange our D_m to be its small-scaled seed dataset and explore new rules from D_u. In **Snowball**'s original setting, firstly two modules are trained on a partial training set. For a new relation, the initial seed data are sampled from test/val set and new seeds are explored from unlabeled data. Since Snowball aims at exploring new knowledge from unlabeled data for a better relation prediction with a little initial seed knowledge, which is similar to SOUP, we compare SOUP with Snowball in our setting. Firstly the two modules are trained on D_m. Then initial seed data are randomly sampled from our test set and new seeds are extracted from D_u.

For **Distant Supervision**, the training set is built by D_u labeled by triplets. To be specific, we train several DSRE models on D_u labeled by different number of Wikidata triplets. Since the triplets are randomly sampled, the results are reported as the average of 5 runs. Notice that all relations are assigned with the same number of triplets.

4.3 Overrall Performance

We evaluate the RE model performance under SOUP mechanism on test set and demonstrate its significant improvement compared with baselines.

The performance of SOUP and baselines are summarized in Table 2. We can see:

(1) Compared with REPEL, SOUP needs much less seed rules to achieve a better performance. This is because REPEL simply takes the shortest dependency paths as rules, which limits the coverage, while SOUP considers various contextual features with flexible weights. Since our rules are more informative, it is reasonable to aggregate the rules' representation for a specific relation to filter out the noisy ones in Rule Filter.

Model		Seed Rules Num.	AUC	Max Micro F1	Max Macro F1	Micro F1	Macro F1	P@100	P@200
REPEL		23.7k	62.05	60.90	55.88	52.97	44.08	100.00	100.00
Snowball		1.6k	25.60	21.44	40.36	9.36	21.34	100.00	100.00
DSRE	BERT-AVG	1.6k	49.04	49.28	39.95	22.19	14.07	99.00	97.00
		3.2k	59.79	57.90	51.98	44.85	33.58	99.00	98.50
		32k	68.54	63.44	58.29	61.64	52.51	100.00	99.50
		72k	71.85	66.33	61.05	66.24	60.08	100.00	100.00
	VAE	1.6k	34.66	41.59	61.16	34.21	61.10	99.00	99.50
SOUP		1.6k	**73.31**	**68.19**	**66.61**	**67.63**	**64.99**	**100.00**	**100.00**

Table 2: Experiment results on our limited seed data settings. Here Seed Rules Num. refers to the size of initial seed rules set. For Snowball, the seed rules is the patterns in initial seeds. For DSRE, the seed rules is Wikidata triplets.

(2) With the same number of initial seed rules, SOUP has a great improvement compared with Snowball, which shows SOUP could explore the knowledge that more helpful for model learning. In Snowball, the knowledge exploration for different relations are independent, while SOUP filters new rules considering the feature of both target relation (POS seed rules) and other relations (NEG seed rules). Thus, SOUP could explore knowledge more distinguishable for the target relation.

(3) By training BERT-AVG with distant supervised labels provided by the different number of triplets (1.6k - 72k), we can find that to get a similar performance as SOUP, DSRE needs nearly 72k triplets and 151k triplet-labeled instances, while SOUP only requires 1.6k initial seed rules and 22k rule-labeled data. This shows that SOUP is good at summarizing informative low-noise knowledge instead of large-scaled high-noise knowledge.

(4) Compared with DSRE-VAE, SOUP achieves a better performance in all metrics with the same number of initial seed rules. This demonstrates the knowledge provided in SOUP's rules explored from unlabeled data is more powerful than that provided by the pre-trained link prediction model.

4.4 Qualitative Analysis of Labeling Noise

To further analyze the labeling noise, we train two BERT-AVG models on the same instances but labeled by our rules (collected after 8 iterations) and 103k Wikidata triplets respectively. Then we evaluate the RE models as agents to estimate the noise in training data. We randomly sample 15k data from D_u for labeling and take the average results of 5 runs. The significant performance gap (9.21% absolute AUC) between DS (57.95% AUC) and SOUP (67.16% AUC) shows that our method brings much less labeling noise than DS.

4.5 Quantitative Analysis of Mutual Enhancement

For DS, the labeling procedure is fixed, while the performance of both logical rules and RE model in SOUP could grow as the exploration on unlabeled data proceeds. To illustrate this, in each step we (1) evaluate the labeling precision of seed rules set on test set and (2) evaluate RE model on test set. The results are shown in Figure 3 and 4. In each step, the new reliable rules explored by RE model power the rules set. Meanwhile, the updated rules could provide training data with lower noise and distinguishable features to learn a better RE model.

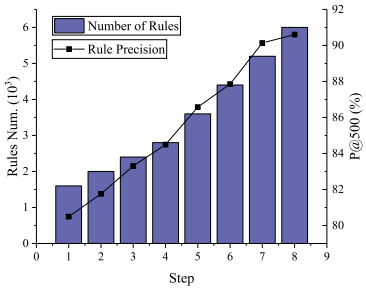

Figure 3: Labeling precision for rules in each step.

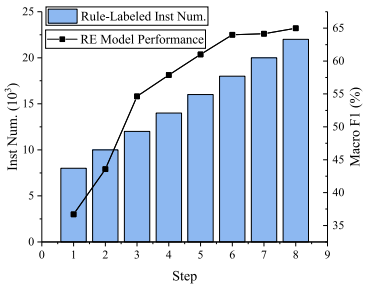

Figure 4: Performance of the RE model in each step.

As the step grows, the improvement of the rule set and RE model becomes smaller. This is because for a target relation, SOUP tends to pick the candidates with semantic feature close to the explored

Step	1	2	3
Overlook hospital is part of atlantic health system which also runs the morristown medical center.			
Label	*member_of*	***subsidiary***	***subsidiary***
The ttc bus route that services most of the length of bayview avenue within toronto is the 11 bayview, which begins at davisville station.			
Label	*occupant*	***location_of_formation***	***location_of_formation***
The spencer davis group was formed in 1963 in birmingham when welsh guitarist spencer davis recruited vocalist and organist steve winwood, and his bass playing brother, muff winwood.			
Label	*instance_of*	*member_of*	***has part***

Table 3: Labels predicted by rules in three steps. The head and tail entities are marked in red and blue respectively. Bold indicates the predicted label is the real label.

knowledge as new seed rules, which stuck SOUP to achieve a complete comprehension for a relation. We plan to investigate this problem in the future.

Case Study We conduct case study to further show our rules work better as the step grows. We collect the rule-labeled results for the same instance provided by different seed rules sets of 1-3 steps, as shown in Table 3. As the step grows, our seed rules set can keep the correct labels in previous steps and predict a relation closer to the real relation for the instance given a wrong label in the past. Therefore, the new seed rules collected in each step are helpful for generating data with lower noise.

4.6 Ablation Study

We conduct ablation studies to investigate the effect of our rule design and Rule Filter.

Specifically, to study the influence of the learned weights and threshold, we conduct SOUP with a variant *Rule-M* (*Appendix A.3 for more details*), whose parameters are fixed as $w_i = 1, i \in \{1, 2, 3\}$ and $TH = 4$. As shown in Figure 5 (a), RE model supervised by rules with learnable parameters consistently outperforms that supervised by Rule-M, demonstrating the effectiveness of the self-adapting weights and thresholds.

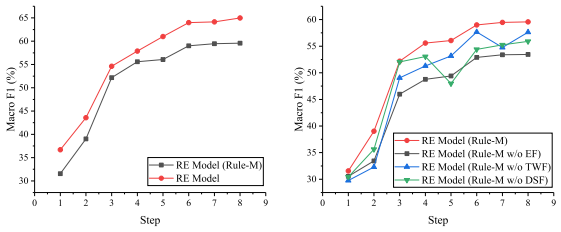

Figure 5: (a) The ablation study on self-adapting weights and threshold. (b) The ablation study on three types of contextual features.

Besides, to study the influence of contextual features considered in rules, we further ablate one type of feature from Rule-M each time (*Appendix A.3 for more details*). The results are shown in Figure 5 (b), where the lack of each type of feature for Rule-M hurts the performance, indicating they can help rules to distinguish relations.

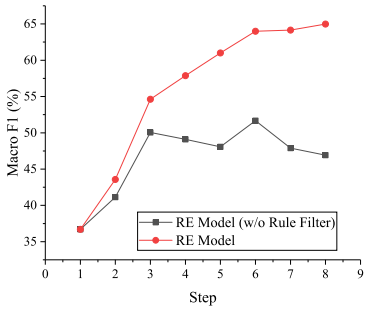

Figure 6: The ablation study on Rule Filter.

Furthermore, to verify the contribution of Rule Filter, we conduct an experiment that in each iteration, directly adds all candidate rules to seed rules set without filtering out noisy rules. As shown in Figure 6, the lack of Rule Filter leads to a much lower RE model performance, which demonstrates this module could denoise the rules to generate more reliable labels for RE training.

5 Conclusion

In this paper, we propose a novel relation extraction framework called SOUP, which continuously explores rules from a large-scale unlabeled dataset to produce low-noise training data. SOUP encourages the collaboration between the RE model and the learnable logical rules. Experimental results show SOUP can generate high-quality rules with a lower requirement of prior knowledge. In the future, we plan to enhance SOUP with the ability to learn rules for unseen relation types automatically.

References

David S Batista, Bruno Martins, and Mário J Silva. 2015. Semi-supervised bootstrapping of relationship extractors with distributional semantics. In *EMNLP*.

Claire Bonial, Julia Bonn, Kathryn Conger, Jena D Hwang, and Martha Palmer. 2014. Propbank: Semantics of new predicate types. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, pages 3013–3019.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2*, pages 2787–2795.

Susan Windisch Brown, Julia Bonn, James Gung, Annie Zaenen, James Pustejovsky, and Martha Palmer. 2019. Verbnet representations: Subevent semantics for transfer verbs. *ACL 2019*, page 154.

Janara Christensen, Stephen Soderland, and Oren Etzioni. 2011. An analysis of open information extraction based on semantic role labeling. In *Proceedings of the sixth international conference on Knowledge capture*, pages 113–120.

Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2021. Distantly supervised relation extraction with sentence reconstruction and knowledge base priors. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 11–26.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Tianyu Gao, Xu Han, Yuzhuo Bai, Keyue Qiu, Zhiyu Xie, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Manual evaluation matters: Reviewing test protocols of distantly supervised relation extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1306–1318.

Tianyu Gao, Xu Han, Ruobing Xie, Zhiyuan Liu, Fen Lin, Leyu Lin, and Maosong Sun. 2020. Neural snowball for few-shot relation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7772–7779.

Xu Han, Tianyu Gao, Yankai Lin, Hao Peng, Yaoliang Yang, Chaojun Xiao, Zhiyuan Liu, Peng Li, Jie Zhou, and Maosong Sun. 2020. More data, more relations, more context and more openness: A review and outlook for relation extraction. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 745–758.

Xu Han, Tianyu Gao, Yuan Yao, Demin Ye, Zhiyuan Liu, and Maosong Sun. 2019. Opennre: An open and extensible toolkit for neural relation extraction. *EMNLP-IJCNLP 2019*, page 169.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke S Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *ACL*.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *ACL (1)*.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. *ACL 2014*, page 55.

Meng Qu, Xiang Ren, Yu Zhang, and Jiawei Han. 2018. Weakly-supervised relation extraction by pattern-enhanced embedding learning. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web,{WWW} 2018,*, volume 2018.

Walid Shalaby, Wlodek Zadrozny, and Hongxia Jin. 2019. Beyond word embeddings: learning entity and concept representations from large scale knowledge bases. *Information Retrieval Journal*, 22(6):525–542.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *EMNLP-CoNLL*.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Xuanhui Wang, Cheng Li, Nadav Golbandi, Michael Bendersky, and Marc Najork. 2018. The lambdaloss framework for ranking metric optimization. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 1313–1322.

Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. 2012. Probase: A probabilistic taxonomy for text understanding. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 481–492.

Jingyi Zhang and Josef van Genabith. 2020. Translation quality estimation by jointly learning to score and rank. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2592–2598.

A Appendix

A.1 Similarity Measurement in Rule Definition

In our rule definition, to decide if an instance x obeys the rule, we apply the methods mentioned in section 3.1.1 to extract the context features and compute the value of $Sim_i(R_{f_i}, x)$ based on the analysis result under i^{th} type of feature.

For entities information (EF), let $pos = \{pos_i\}_{i=1}^2$ be the part of speech and $e^c = \{e_i^c\}_{i=1}^2$ be the concept vectors.

For trigger words information, let $srl = \{srl_i\}_{i=1}^2$ be semantic roles, $trl = \{trl_i\}_{i=1}^2$ be thematic roles, $\mathcal{E} = \{s_t, v_t\}_{t=1}^T$ be the event information expressed in the sentence (s_t donates the event state, v_t donates the event words and T is the length of events). Figure 7 shows an example. The trigger words representation extracted by OpenIE and BERT is e^v.

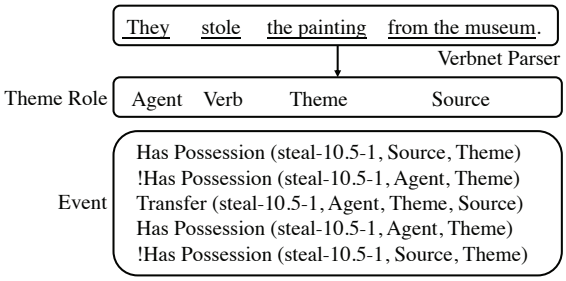

Figure 7: Verbnet result for the sentence "They stole the painting from the museum".

For dependency structure, Figure 8 shows an example. The path between e_1 and e_2 is $\mathcal{P} = \{P, U, T\}$, where $P = \{p_i\}_{i=1}^k$ is the POS tag of nodes in the path, $U = \{u_i\}_{i=1}^{k-1}$ and $T = \{t_i\}_{i=1}^{k-1}$ are the direction and type of edges. ($u_i \in \{forward, back\}$)

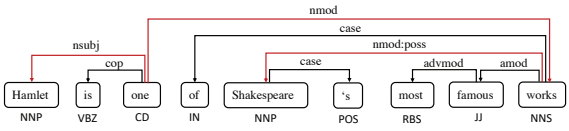

Figure 8: For sentence "Hamlet is one of Shakespare's most famous works", the red line marks shortest dependency path between *Hamlet* and *Shakespeare*, which is "NNP (Hamlet) <back, nsubj> CD(one) <forward, nmod> NNS(works) <forward, nmod:poss> NNP(Shakespeare)".

For the i^{th} type of feature, $Sim_i(R_{f_i}, x)$ outputs a similar score between x and R_f on this feature. To be specific, $Sim_i(R_{f_i}, x) = \sum_{p=1}^{k_i} sub\_sc_i^p(R_{f_i}, x)$, where $sub\_sc_i^p(\cdot,\cdot)$ is the similarity score for the p^{th} analysis result in i^{th} type of feature R_{f_i} (as shown in Table 4-6). k_i is the number of analysis methods for i^{th} type of feature in R_f. ($k_1 = 3$, $k_2 = 4$ and $k_3 = 1$)

Criterion	value
$pos^{(R_{f_1})} = pos^{(x)}$	$sub\_sc_1^1 = 1$
$cosSim(e_1^{c(R_{f_1})}, e_1^{c(x)}) > 0.7$	$sub\_sc_1^2 = 1$
$cosSim(e_2^{c(R_{f_1})}, e_2^{c(x)}) > 0.7$	$sub\_sc_1^3 = 1$

Table 4: Definition for $sub\_sc_1(R_{f_1}, x)$. $cosSim$ donates cosine similarity.

Criterion	value
$srl^{(R_{f_2})} = srl^{(x)}$	$sub\_sc_2^1 = 1$
$trl^{(R_{f_2})} = trl^{(x)}$	$sub\_sc_2^2 = 1$
$\mathcal{E}_{same} \geq \lvert\mathcal{E}^{(R_{f_2})}\rvert / 2$	$sub\_sc_2^3 = 2$
$cosSim(e^{v(R_{f_2})}, e^{v(x)}) > 0.7$	$sub\_sc_2^4 = 1$

Table 5: Definition for $sub\_sc_2(R_{f_2}, x)$. $\lvert\mathcal{E}^{(R_{f_2})}\rvert$ donates the number of elements in $\mathcal{E}^{(R_{f_2})}$. $\mathcal{E}_{same}$ donates the number of same elements between $\mathcal{E}^{(R_{f_2})}$ and $\mathcal{E}^{(x)}$

Criterion	value
$\mathcal{P}^{same} = \mathcal{P}^{(R_{f_3})}$	$sub\_sc_3^1 = 2$
$\lvert\mathcal{P}^{same}\rvert \geq \lvert\mathcal{P}^{(R_{f_3})}\rvert/2$ and $\lvert\mathcal{P}^{same}\rvert \geq \lvert\mathcal{P}^{x}\rvert/2$	$sub\_sc_3^1 = 1$

Table 6: Definition for $sub\_sc_3(R_{f_3}, x)$. $\mathcal{P}^{same}$ is the sub-path shared by both $\mathcal{P}^{(R_{f_3})}$ and $\mathcal{P}^{(x)}$. $\lvert\mathcal{P}_{R_{f_3}}\rvert$ is the length of $\mathcal{P}^{(R_{f_3})}$ and $\lvert\mathcal{P}^x\rvert$ is the length of $\mathcal{P}^{(x)}$.

A.2 Hyperparameters and Training Details

Most of the hyperparameters are listed in Table 7. For optimization, Adam (initial learning rate = 0.001) is applied to train the ranking model in Rule Generator and rules classifier in Rule Filter, while SGD (initial learning rate = 0.1) is employed to train our RE model. The epoch number of the ranking model, rules classifier and RE model is 100, 200 and 5. The bag size of RE model is 4. Each experiment contains 8 iterations.

In each iteration, 30k data are sampled from Du and the top 8k or 2k rule-labeled ones with the highest Sc_{sim} are added to the training set for RE model.

Component	Parameter	Value
Rule Generator	Threshold for single-feature rules	TH={2, 3, 1}
	Rank model's hidden size	768
	Relation embedding dimension	512
	Rank-to-Weight list	r2w={2, 1.5, 1}
	Candidate threshold set	THS={6, 7, 8}
Rule Filter	Rule embedding dimension	768
	GAT's multi-head number	3
	Output node representation dimension	64
	Number of initial seed rules (per relation)	N_I=20
	Number of candidate rules (per relation)	N_C=50
	Number of new positive seed rules (per relation)	N_S=5 (iteration 1-4); N_S=10 (iteration 5-8)
RE Model Training	Number of sampled unlabeled data	N_u=30k
	Number of data added to training set	N_t=8k (iteration=1); N_t=2k (iteration=2-8)

Table 7: Hyper-parameters of the components in our experiments.

Rule	Definition
Rule-M	$isObey_R(x) = \sum_{i=1}^{\|R_f\|} Sim_i(R_{f_i}, x) \geq 4, R_f = \{EF, TWF, DSF\}$
Rule-M (w/o EF)	$isObey_R(x) = \sum_{i=1}^{\|R_f\|} Sim_i(R_{f_i}, x) \geq 3, R_f = \{TWF, DSF\}$
Rule-M (w/o TWF)	$isObey_R(x) = \sum_{i=1}^{\|R_f\|} Sim_i(R_{f_i}, x) \geq 3, R_f = \{EF, DSF\}$
Rule-M (w/o DSF)	$isObey_R(x) = \sum_{i=1}^{\|R_f\|} Sim_i(R_{f_i}, x) \geq 3, R_f = \{EF, TWF\}$

Table 8: Rules definition in ablation study.

A.3 Rule Definition in Ablation Study

Table 8 introduces the definition of the rules in ablation studies.