Distilling Task-specific Logical Rules from Large Pre-trained Models

Anonymous ACL submission

Abstract

Logical rules, both transferable and explainable, are widely used as weakly supervised signals for many downstream tasks such as named entity tagging. To reduce the human effort of writing rules, previous researchers adopt an iterative approach to automatically learn logical rules from several seed rules. However, obtaining more seed rules can only be accomplished by extra human annotation with heavy costs. Limited by the size and quality of the seed rules, the model performance of previous systems is bounded. In this paper, we develop a novel framework STREAM to distill task-specific logical rules from large pre-trained models. Specifically, we borrow recent prompt-based language models as the knowledge expert to yield initial seed rules, and based on the formed high-quality instance pool that acts as an intermediary role, we keep teaching the expert to fit our task and learning task-specific logical rules. Experiments on three public benchmarks demonstrate the effectiveness of our proposed framework. Without any participation of manual annotation, our system has gained significant improvements over previous state-of-the-art methods.

1 Introduction

Following the supervised learning paradigm, researchers resort to human annotation to obtain training data for specific tasks such as named entity tagging and relation extraction. Though accurate, manually annotated data construction is quite expensive and time-consuming. In real scenarios, logical rules often serve as a source of weak supervision that provides abundant weakly supervised data for various downstream models, and compared with labeling data, applying rules can cover more application domains with better interpretability. Therefore, rule-based labeling systems (Figure 1) have attracted considerable attention in recent years.

In fact, it's not easy to develop an accurate and complete rule system, as the logical rules are usually summarized by human experts and the building process requires extensive domain knowledge. Besides, there is no evaluation metric to guide annotators to select valuable rules. The usability and quality of acquired rules can not be guaranteed. In this sense, how to build a reliable rule system with limited human effort is still an important challenge.

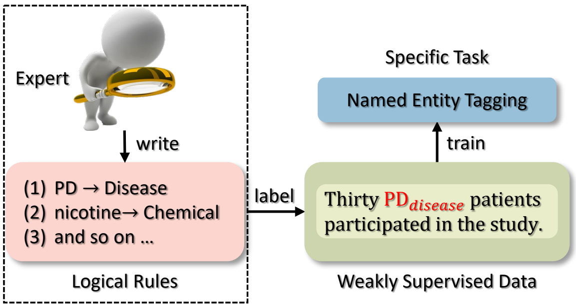

Figure 1: Schematic diagram of a rule-based weakly supervised named entity tagging system. Our goal in this work is to learn logical rules without any human effort, corresponding to the dotted area in the figure.

To solve above issue, previous researchers pay attention to the automatic construction of logical rules, which tends to start from a few seed rules and learn new logical rules by pre-defined similarity measures in an iterative manner. Though proven to be effective, these systems still require manually constructed seed rules as the cold start. Limited by human effort, the size of seed rules is usually small so that the system performance is bounded.

In this work, we propose a fully automated framework STREAM to diStill Task-specific logical Rules from large prE-trAined Models, and this framework can take human out of the loop indeed. Specifically, (1) in order to get rid of the restrictions of the seed rules, we firstly ask large pre-trained models for help. As the prompt-based pre-trained models own the zero-shot ability to generate candidate entity types, we design two appropriate prompt templates and achieve automatic acquisition of seed rules by the model outputs' consistency.

(2) Once seed rules are obtained, we form a high-quality instance pool to continuously add potential instances to the pool and distill new logical rules from the pool in an iterative manner. (3) At last, based on the convergent instance pool, we further fine-tune a new prompt-based model with more suitable prompt templates to obtain more reliable seed rules. Notably, each step in the framework does not require any human involvement.

Experiments on three public named entity tagging benchmarks demonstrate the effectiveness of our proposed framework STREAM, with consistent improvements over several baseline models and far exceed the state-of-the-art (SOTA) systems. Besides, we perform a detailed ablation study to analyze the quality of our obtained seed rules, the convergence of our propose iterative framework, and some specific cases of learned logical rules.

Accordingly, the major contributions of our work are summarized as follows:

(1) We introduce the large pre-trained prompt-based models to end the dilemma that the logical rule learning systems require seed rules as a start.

(2) We develop an effective and stable framework to distill logical rules in an iterative manner, which combines prompt-based fine-tuning and rule distillation to achieve mutual enhancement.

(3) We conduct detailed experiments to illustrate the effectiveness and rationality of our framework — without any human involvement, the performance of STREAM has surpassed previous rule learning systems based on manually selected seed rules.

2 Methodology

2.1 Overview

In this work, we adopt named entity tagging as the specific downstream task to compare with previous work (Li et al., 2021) of learning logical rules. The diagram of STREAM is visualized in Figure 3.

2.2 Logical Rules

In real scenarios, logic rules can appear in various forms. For convenience, we define the logical rules in the unified form of *"if p then q (i.e.p $\rightarrow$ q)"*. In named entity tagging task, *"p"* can be any logical expression and *"q"* is the corresponding entity category. For example, a logical rule may look like: "*if the entity's lexical string is PD*[1]*, then its corresponding entity label should be* ***disease***".

[1] PD: Parkinson's disease

As demonstrated in previous work (Zhou and Su, 2002), we define five meta logical rules to tag named entities based on their lexical, contextual, and syntax information. In addition, some combinations of simple logical rules are also considered.

2.2.1 Meta Logical Rules

Following existing literature, our pre-defined meta-rules are: (1) TokenString rule matches entity's lexical string; (2) PreNGram rule matches entity's preceding context tokens; (3) PostNGram rule matches entity's succeeding context tokens; (4) POSTag rule matches entity's part-of-speech tags; (4) DependencyRel rule matches the dependency relations of the entity and its headword.

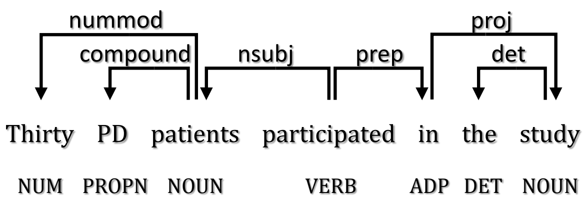

Figure 2: Dependency parsing example.

Figure 2 shows an example with its dependency structure. In this sentence, word *PD* is a potential disease entity and following logical rules may exist:

TokenString == *PD* $\rightarrow$ disease
PreNGram == *thirty* $\rightarrow$ disease
PostNGram == *patients* $\rightarrow$ disease
POSTag == PROPN $\rightarrow$ disease
DependencyRel ==
(compound, *patient*) $\rightarrow$ disease

In fact, above simple rules may sometimes fail to work, therefore we introduce complex rules, which combine several simple rules into compound rules by logical connectives including and ($\wedge$), or ($\vee$) and negation ($\neg$). For example, only a mention that satisfies both rule PostNGram == *patients* and rule POSTag == PROPN can be a disease entity.

2.2.2 Logical Rules Mining

After defining the form of meta logical rules, we traverse the entire training set and recall all potential rules that satisfy the format of meta rules.

2.3 Zero-shot Prompt Models as Seed Rules

In our proposed framework STREAM, we design a zero-shot prompt-based module to generate seed rules from pre-trained models without any human participation, and the details are as follows.

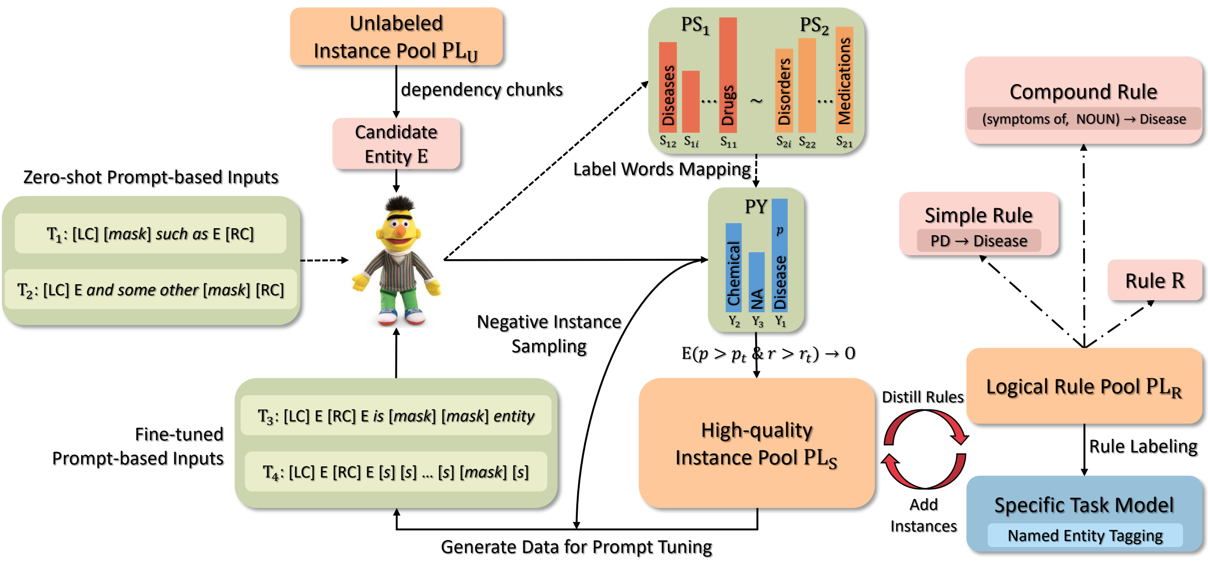

Figure 3: (1) In the first loop, zero-shot prompt-based models provide initial seed rules (*i.e.*dotted line) and form the high-quality instance pool PL_S. (2) In the following loops, STREAM uses the positive instances from PL_S and negative instances from sampling to fine-tune prompt-based models, generate better seed rules, and form a new instance pool PL_S. (3) During the process, simple and compound logical rules are distilled from the pool PL_S.

2.3.1 Zero-shot Prompt Model Inputs

For each sentence in the unlabeled training corpus PL_U, we first obtain its noun chunks E by open-source dependency parsing tools such as *Spacy*[2]. Actually, each noun chunk E_i may be a potential entity so that we construct the following prompt-based input to mine its possible entity types:

$$T_1 : LC \ldots [mask] \textit{ such as } E \ldots RC \quad (1)$$

where LC and RC are left and right context tokens. With the help of the zero-shot capability of large pre-trained models, we can obtain the condition probability output ($PS \subset \mathbb{R}^V$, where V is the vocabulary size) of noun chunk E at the position of special token [*mask*]. For example, the prompt input of the above sentence *"Thirty PD patients participated in the study"* with noun chunk *PD* is:

$$\textit{Thirty } [mask] \textit{ such as PD patients} \ldots \textit{study} \quad (2)$$

At the position of word [*mask*], pre-trained models are able to directly output entity types like *"diseases"*, *"disorders"*, *"conditions"* and so on, and we select *Top-K* types ($S_1 : \{S_{11}, S_{12}, \ldots, S_{1K}\}$) from the output condition probability $PS_1 : PS$ as the candidate types of noun chunk E_i, where S_{1i} is the entity type that outputs the i_{th} highest confidence in the conditional probability PS_1.

However, the above prompt-based input breaks left context and focus more on right context, and we propose another prompt-based input to take left context also into consideration. It is defined as:

$$T_2 : LC \ldots E \textit{ and some other } [mask] \ldots RC \quad (3)$$

For this prompt-based input, we can also obtain its *Top-K* entity types ($S_2 : \{S_{21}, S_{22}, \ldots, S_{2K}\}$), and the final type set S of noun chunk E should be generated from two above type sets S_1 and S_2.

2.3.2 Label Words Mapping

To bridge the gap between model's output types S and our task-specific categories Y, we design a module to find label words mapping automatically.

Specially, to find the label word mapping for target category Y_i, we count the co-occurrence between prompt model's output types S_i and target type Y_i in all S, and filter out the types with high support ($S_{ih} : S$). Actually, the co-occurring types in S_{ih} are often synonyms or aliases of the target type Y_i. For instance, we can find target category *diseases* and type *disorders* tend to appear together in S, which means there is a label word mapping: *disorders* $\to$ *disease*. We define the founded label mapping as $\mathcal{M}$, which maps all entity types in S_{ih} to target type Y_i, this is defined as:

$$\mathcal{M} : S_{ih} \to Y_i, \; Y_i \in Y \quad (4)$$

[2] https://spacy.io/models/en#en_core_web_sm

2.3.3 Zero-shot Seed Rules

By the label word mapping $\mathcal{M}$, we can convert the initial condition probability output $\text{PS}_{1,2}$ to our task-specific condition probability $\text{PY}_{1,2} \subset \mathbb{R}^v$, where v is the target category number.

PY_1 and PY_2 are the entity type predictions for noun chunk E that come from two different prompt-based models. Therefore, if the two models' predictions are similar and have no differences, the final entity type O can be considered as the common output type of the two models. It is defined as:

$$\text{O} = \begin{cases} \text{Y}_{11} & \textit{if} \quad \text{Y}_{11} = \text{Y}_{21}, \\ \text{unk} & \textit{otherwise.} \end{cases} \tag{5}$$

where Y_{1i} is the entity type with the i_{th} largest model confidence p_{1i} in PY_1, and unk means the entity type is unknown due to models' divergence. For these chunks E with determined entity type $\text{O} \neq \text{unk}$, we further filter out the chunks with high model confidence p, where $p = \min(p_{11}, p_{21})$.

Besides, we also use a support threshold to further filter out high-quality chunks (*i.e.*chunks that occur less often may be noisy), and we define the final obtained chunks pool as PL_S.

$$\text{PL}_S = \{\text{E} : p > p_t, r > r_t\} \tag{6}$$

where p_t, r_t are confidence and support thresholds. Actually, any noun chunk E with entity category O in the high-quality chunk pool PL_S can be seen as an initial TOKENSTRING rule:

$$\text{TOKENSTRING} == \text{chunk E} \rightarrow \text{type O} \tag{7}$$

2.4 Distill Task-specific Logical Rules from High-quality Instance Pool

Once seed rules are obtained, we can fetch all instances that matches the seed rules to form a instance pool $\text{PL}_S \rightarrow \text{PL}_R$. Based on the initial pool PL_R, we aim to add high-quality instance to the pool, and distill new logical rules from the pool.

2.4.1 Add High-quality Instances to the Pool

Based on the high-quality instances in PL_S, we can train specific (*i.e.*named entity tagging) downstream models, and the trained model is defined as $\mathcal{F}$. After that, we use the trained model to generate a pseudo label for each unlabeled instance in PL_U.

To identify potential high-quality instances in unlabeled instances PL_U , we use the instances in high-quality pool PL_S as a guide. In detail, for any unlabeled sentence $s_u \in \text{PL}_U$, its pseudo label given by the trained model is $\text{Y}_u = \mathcal{F}(s_u)$. We randomly sample a certain number of high-quality instances from PL_S with the same entity label Y_u, and estimate the similarity between the instance s_u and these sample instances. This is defined as:

$$\text{S-score}(s_u) = \text{Medium}[\text{sim}(s_u, s_i)], \; s_i \in \text{PL}_S \tag{8}$$

where sim is the function to measure the semantic similarity between sentence s_u and the sampled sentence s_i, and Medium means that the final score is the median (*i.e.*avoid the influence of outliers) of all pair scores ($\text{sim}(s_u, s_i), s_i \in \text{PL}_S$). Besides, to decide the score threshold of adding instances to the pool PL_S, we also randomly select instance s_j from PL_S, and calculate the similarity score between the instance s_j and the remaining instances PL_S/s_j. This process is defined as:

$$\text{S-score}_t = \text{Medium}[\text{S-score}(s_j, \text{PL}_S/s_j)], \; s_j \subset \text{PL}_S \tag{9}$$

2.4.2 Distill Task-Specific Rules from the Pool

With such a high-quality instance pool PL_S, our goal is to find all high-quality rules from potential rules set R mined in section 2.2.2. For any rule $\text{R}_u \in \text{R}$, we define its confidence score as:

$$\text{R-score}(\text{R}_u) = \frac{M_{\text{R}_u}}{N_{\text{R}_u}} \log_2 N_{\text{R}_u} \tag{10}$$

where N_{R_u} is the number of sentences that meets rule R_u in the pool PL_S, and M_{R_u} is the number of sentences that matches rule R_u correctly (*i.e.*the rule labelling result is consistent with the high-confidence label O). Similarly, we also use the existing high-quality rules to determine the dynamic threshold for filtering out potential rules:

$$\text{R-score}_t = \text{Medium}[\text{R-score}(\text{R}_i)], \; \text{R}_i \subset \text{PL}_R \tag{11}$$

Accordingly, we keep repeating the steps defined in sections 2.4.1 and 2.4.2 until pool PL_S or PL_R is no longer updated. During this process, high-quality instances are gradually added to the instance pool PL_S, and corresponding high-quality logical rules are also produced.

2.5 Fine-tuned Prompt Model as Seed Rules

In section 2.3.1, we propose to utilize zero-shot prompt-based models to generate initial seed rules, however, these rules are just a compromise at the time of data cold start (*i.e.*w/o any weakly labeled data). Once we have collected enough high-quality instances in pool PL_S, we can further adjust the prompt-based model to adapt to our specific task, and generate seed rules with higher quality.

2.5.1 Fine-tuned Prompt Model Inputs

To further fine-tuned prompt models, we construct two new prompt-based inputs as follows:

$$\begin{aligned} &T_3: \text{LC ... E ... RC E } \textit{is [mask] [mask] entity} \\ &T_4: \text{LC ... E ... RC E } [s]\ [s] \ldots [s]\ [\textit{mask}]\ [s] \end{aligned} \quad (12)$$

where [*s*] is the soft mask token. In the input template T_3, we aim to make the prompt-based model to output entity types at the positions of two [*mask*] tokens, and its label words mapping is:

$$\begin{array}{cccl} [\textit{mask}] & [\textit{mask}] & & \\ \textit{a/an} & \text{O} & \rightarrow & \text{E}\textit{'s entity label is } \text{O} \\ \textit{not} & \textit{an} & \rightarrow & \text{E } \textit{is not an entity} \end{array} \quad (13)$$

For example, if the output words at the positions of two mask tokens [*mask*] are "*a disease*", it means chunk E is an entity and its label is disease.

Besides, we also use a soft embedding prompt-based input scheme (*i.e.*T_4), which can implicitly learn word embeddings at the positions of [*s*] through gradient propagation. In this case, our goal is to constrain the model to output the target entity types O directly at the position of [*mask*].

Compared to the zero-shot prompt-based input proposed in section 2.3.1, the above two prompt-based inputs do not destroy the original sentence structure and promote the models to better understand the meaning of the entire sentence.

2.5.2 Negative Instance Sampling

However, instances in pool PL_S are all positive sentences so that the models can not be trained only on pool PL_S. To solve this issue, we sample some high-quality negative instances also based on the consistency of model outputs, this is defined as:

$$O = \begin{cases} Y_{11} & \textit{if} \quad Y_{11} = Y_{21}, \\ \text{NA} & \textit{otherwise if} \quad S_{11} = S_{21}, \\ \text{unk} & \textit{otherwise.} \end{cases} \quad (14)$$

In short, negative samples (*i.e.*NA) are the samples with same zero-shot prompt model outputs ($S_{11} = S_{21}$), but not with any target entity type ($S_{11} \not\subset Y$).

2.5.3 Fine-tuned Seed Rules

Based on the positive sentences provided by the pool PL_S and negative instances sampled in section 2.5.2, we can fine-tune prompt-based models with the inputs defined in section 2.5.1. Then, similar to the approach in section 2.3.3, we use the fine-tuned models to predict pseudo labels for all unlabeled sentences, and filter out the sentences with high model confidence and high support as new seed rules. Immediately afterward, our system will repeat the steps in sections 2.4 and 2.5 to continuously distill more new rules, form a larger pool PL_S and fine-tune a better prompt-based model.

3 Experiments

Our experiments are designed to verify the effectiveness of our proposed system — STREAM.

3.1 Benchmark

BC5CDR (Li et al., 2016) is constructed with BioCreative VCDR task corpus. It contains 500 train, 500 dev and 500 test PubMed articales, with 15,953 chemical and 13,318 disease entities.

CHEMDNER (Krallinger et al., 2015) contains 10,000 PubMed abstracts with 84,355 chemical entities, in which the training/dev/test set contain 14,522/14,572/12,434 sentences respectively.

CONLL2003 (Tjong Kim Sang, 2002) consists of 14,041/3,250/3,453 sentences in the training/dev/test data split of Reuters news articles[3].

3.2 Model and Metric

To evaluate the quality of our generated rules, we use the rule-labeled data to train the named entity tagging model and report corresponding model performance. The evaluation metrics in our experiments include Precision(P), Recall(R) and F1-score(F_1). In STREAM, we use the model in (Jiang et al., 2020) as the specific tagging model.

3.3 Baseline

We select several recent weakly supervised methods to compare, including current SOTA systems.

Seed Rules uses manually annotated seed rules to match test set and evaluate the label performance.

Seed-Tagger uses seed rules to label test set and train the tagging models on the labeled data.

[3]Following previous work, type MISC is not considered.

Method	Need Seed ?	BC5CDR			CHEMDNER*			CONLL2003		
		P	R	F_1	P	R	F_1	P	R	F_1
Seed Rules	✓	**94.09**	3.81	7.33	**91.60**	13.19	23.07	**95.77**	2.76	5.36
Seed-Tagger	✓	78.33	21.60	33.86	84.18	21.91	34.78	72.57	24.68	36.83
LinkedHMM	✓	10.18	15.60	12.32	23.99	10.77	14.86	19.78	31.51	24.30
HMM-Agg	✓	43.70	21.60	29.00	49.60	18.40	26.80	52.00	8.50	14.60
CGExpan	✓	40.96	24.75	30.86	45.70	25.58	32.80	55.97	28.70	37.95
AutoNER	✓	42.22	30.66	35.52	66.83	27.59	39.05	32.07	5.98	10.08
Self-Training	✓	73.69	29.55	42.19	85.06	20.03	32.42	72.80	24.83	37.03
TALLOR	✓	66.53	66.94	66.73	48.34	52.56	50.36	64.29	64.14	64.22
STREAM	✗	72.47	**67.90**	**70.11**	63.93	**55.13**	**59.20**	69.92	**72.30**	**71.09**

Table 1: Model performances on BC5CDR, CHEMDNER, and CONLL2003. Bold and underline indicate the best and the second best scores, * means the reported result is our re-implementation of author-provide code.

CGExpan (Zhang et al., 2020) expands lexicons by language models. Following previous work, we use CGExpan to expand the size of human annotated TokenString rules (*i.e.*lexicons) to 1000.

AutoNER (Shang et al., 2018b) labels untyped terms automatically with a pre-defined dictionary. We use the best expanded lexicon from CGExpan as the dictionary. Both of the expanded lexicon and the mined phrases from AutoPhrase (Shang et al., 2018a) as untyped mined phrases.

LinkedHMM (Safranchik et al., 2020) proposes to utilize a generative model to aggregate noisy rules, and forms weak supervision signals to train the models. We use the best expanded lexicon from CGExpan as the tagging rules and the mined phrases from AutoPhrase as the linking rules.

HMM-Agg (Lison et al., 2020) introduces the hidden Markov models to generate weak labels by labeling functions. We use the best expanded lexicons from CGExpan as the labeling functions.

Self-Training uses the seed rules to get initial teacher models and iterates the processes of generating pseudo labels for unlabeled data and training student models following self-training scheme.

TALLOR (Li et al., 2021) bootstraps high-quality logical rules to train a neural tagger in an iterative manner, with selected, the most frequent manually annotated seed rules as the input.

3.4 Overall Performance

We summarize the model performances of our STREAM and above mentioned baselines in Table 1. From the table, we can see: (1) Method Seed Rules yield a high accuracy, however, this simple matching pattern lacks generalization ability and results in a low model recall. (2) Similarly, the Self-training method starts from a small amount of seed data and has a good model accuracy, but its model recall is poor due to the limited data size. (3) Lexicon expanded model CGExpan and AutoNER sacrifice a certain model accuracy in exchange for more balanced model performance. (4) Previous SOTA system TALLOR can learn logical rules in an iterative manner and achieves competitive model F1-score. Since this system still relies on initial seed rules, its model performance is bounded.

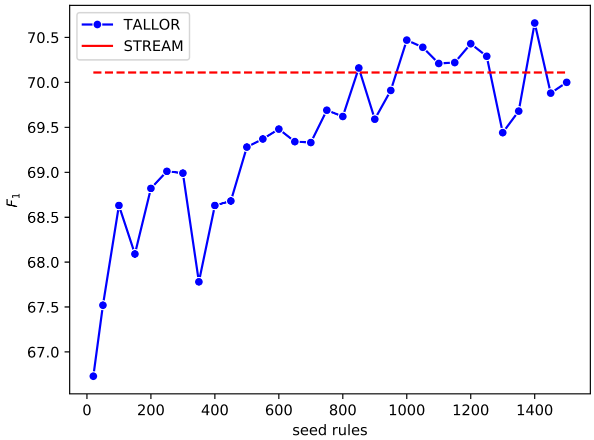

Figure 4: Model Performances on dataset BC5CDR with different manually annotated seed rules size.

Compared to the above baseline models, our proposed system STREAM does not need any human-annotated seed rules or data. We draw curves in Figure 4 to illustrate how many seed rules TALLOR needs to be comparable to our STREAM. From the figure, we can see that: When the number of most frequent seed rules reaches 825, the model performance of TALLOR exceeds STREAM for the first time. However, acquiring such the most frequent seed rules is quite labor-intensive.

3.5 Ablation Study

To explore how our proposed system STREAM works, we now present ablation studies.

Initial Seed Rules In section 2.3.3, we utilize zero-shot prompt-based models' consistent outputs as initial seed rules. We firstly conduct experiments to check the quality of initial seed rules.

Method	BC5CDR			CHEMDNER		
	P	R	F_1	P	R	F_1
TALLOR	66.53	66.94	66.73	48.34	52.56	50.36
STREAM_{init}	70.46	64.99	67.62	57.13	50.52	53.62
STREAM	72.47	67.90	70.11	63.93	55.13	59.20

Table 2: Ablation results of different rule learning systems on BC5CDR and CHEMDNER, method STREAM_{init} uses the initial seed rules.

We summarize the model performances of different logical rule learning systems in Table 2. From the table we can see that: With only the initial rules given by the zero-shot prompt model, our system STREAM_{init} has surpassed the previous SOTA method TALLOR in metrics P and F_1, which means the generated seed rules are reliable. Besides, we directly utilize the human-annotated labels to verify the quality of the initial rules in a more intuitive way: When confidence threshold $p_t = 0.3$ and support threshold $r_t = 4$, about 219 seed rules are obtained with an accuracy of 98.6%.

In the process of initial seed rules generation, hyperparameters p_t and r_t are relatively important. We draw the figure in Figure 5 to show the model performances of STREAM_{init} with different combinations of p_t and r_t. From the figure, we can see that: (1) When $p_t = 0.3$ and $r_t = 0.4$, our system STREAM_{init} achieves the best performance. (2) As hyperparameters p_t or r_t increases, the model performance first increases and then decreases. This is because a low parameter value may introduce some data noise, while a high parameter value may reduce the number of recalled rules.

Fine-tuned Seed Rules After fine-tuning prompt-based models, the model can output high confidence scores for those correct samples. Thus, to filter out them, we adopt p_t as 0.99, and following previous experience, we adopt r_t as 4. With this combination, we can find that about 272 seed rules are recalled with an accuracy of 97.4%.

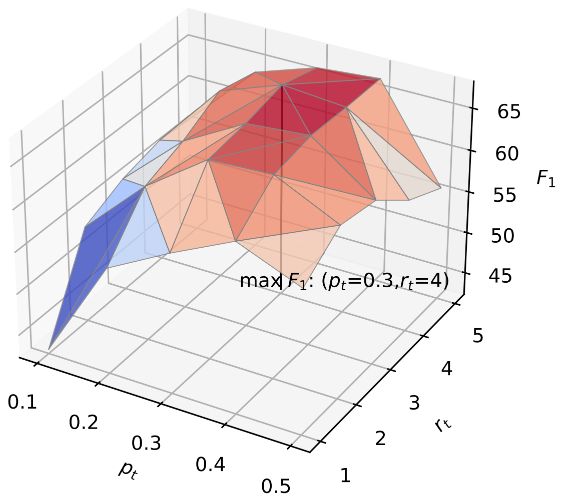

Figure 5: Model performances of different hyperparameter combinations on dataset BC5CDR.

In section 2.5.1, we propose two prompt-based inputs T_3 and T_4: T_3 directly uses hard words to form the prompt sentence while T_4 introduces soft embeddings to learn during training process. To compare the two prompt inputs, we present an ablation study, and the results are in Table 3.

Method	BC5CDR			CHEMDNER		
	P	R	F_1	P	R	F_1
TALLOR	66.53	66.94	66.73	48.34	52.56	50.36
STREAM_{hard}	73.13	65.82	69.28	61.49	55.65	58.42
STREAM	72.47	67.90	70.11	63.93	55.13	59.20

Table 3: Ablation results of different prompt-based inputs on BC5CDR and CHEMDNER, method STREAM_{hard} uses the prompt-based input T_3, method STREAM uses the prompt input T_4.

From the table we can see that: (1) Whether template T_3 or T_4 is used, our system STREAM can achieve SOTA model performance. (2) Soft template T_4 performs better because it can learn a more efficient prompt pattern during training process.

High-quality Instance Pool In section 2.4, we add high-quality instances to PL_S and distill new logical rules from PL_S in an iterative manner. To figure out how PL_S changes during the training process, we now show the ablation in Figure 6.

From the figure, we can see: (1) During the iterative process, the size of PL_S continues to increase, but its growth rate gradually slows down. (2) At the beginning of the iteration, high-quality instances are added to the pool PL_S, and the model performance increases. However, at the later itera-

Logical Rule Matched Sentence	Logical Rule Condition p	Entity Type q
Other **symptoms of** **allergic$_{\text{ADJ}}$ reactions$_{\text{NOUN}}$** were not clinically detectable.	PRENGRAM == **symptoms of** $\wedge$ POSTAG == **ADJ NOUN**	$p \rightarrow q$: **disease**
Grade less than or equal to 2 **nausea$_{\text{NOUN}}$** **and vomiting** occurred in 66%, courses and phlebitis ...	POSTNGRAM == **and vomiting** $\wedge$ POSTAG == **NOUN**	$p \rightarrow q$: **disease**
This study describes neuropsychiatric side effects in patients **after treatment with** **mefloquine$_{\text{PROPN}}$**.	PRENGRAM == **after treatment with** $\wedge$ POSTAG == **PROPN**	$p \rightarrow q$: **chemical**
Prophylactic use of lamivudine with chronic immunosuppressive **therapy for** **rheumatologic disorders.**	PRENGRAM == **therapy for** $\wedge$ POSTNGRAM == **[END]**	$p \rightarrow q$: **disease**

Table 4: Cast study of learned logical rules on dataset BC5CDR, [END] means the end (.) of sentences.

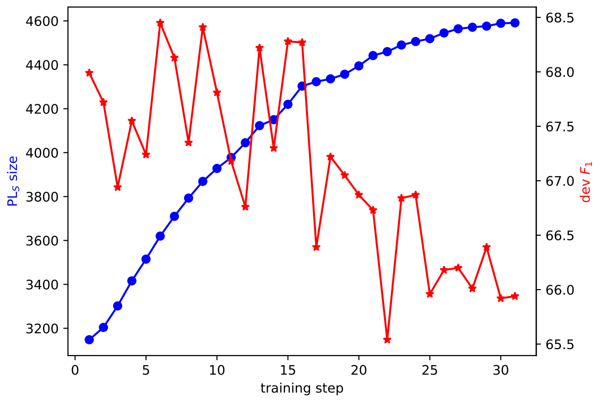

Figure 6: Changes of PL$_S$ size and model performance on development set (*i.e.*dev F_1) during training process.

tion stage, some noise instances may be introduced, which causes the model performance to decrease.

3.6 Case Study: Learned Logical Rules

In the above section, we have proved that our system STREAM can extract better logic rules with better downstream model performance. In this section, we present a case study to show the learned logical rules from a more intuitive perspective.

In Table 4, we select four sentences and corresponding learned logical rules from the training corpus. For example: (1) The entity mention in the first sentence matches the POSTAG form of **ADJ NOUN**, and its PRENGRAM words are **symptoms of**, therefore, STREAM can infer a rule: (PRENGRAM == **symptoms of** $\wedge$ POSTAG == **ADJ NOUN**) $\rightarrow$ **disease**. (2) In the forth sentence, the PRENGRAM words of mention **rheumatologic disorders** are **therapy for** while the mention is just the end of sentence. STREAM can extract logical rule (PRENGRAM == **therapy** for $\wedge$ POSTNGRAM == **END**) $\rightarrow$ **disease** from this sentence.

4 Related Work

Weak Supervision To alleviate the issue of limited labeled data, previous researchers made many efforts to improve *named entity tagging* systems from different perspectives: (1) (Ren et al., 2015; Fries et al., 2017; Giannakopoulos et al., 2017) introduce distant supervision (Mintz et al., 2009), an automated method to label data by aligning text with remote knowledge bases, to build NER systems without human supervision. (2) (Shang et al., 2018a) uses typed lexicons and (Peng et al., 2019) uses incompetent dictionaries as the indirect supervision to guide model training. (3) Recently, (Li et al., 2021) proposes to learn logical rules from selected seed rules to generate more diverse pseudo labels, and achieves the SOTA model performance. However, above systems still rely on seed data or rules, so that the model performance is bounded.

Language Models Vaswani et al. (2017) proposed a self-attention based architecture — Transformer, and it soon becomes the backbone of many following language models. By pre-training on a large-scale corpus, BERT (Devlin et al., 2019) obtains the ability to capture a notable amount of "common-sense" knowledge and gains significant improvements on many tasks following the fine-tune scheme. Recently, (Gao et al., 2021; Han et al., 2021; Wei et al., 2021) found that the prompt-based models achieve remarkable few-shot performance, and reformulate the traditional paradigm of fine-tuning to prompt-tuning, which could better utilize the knowledge of the pre-trained models.

5 Conclusion

In this work, we propose an automated framework STREAM to distill task-specific logical rules from large pre-trained models. Experiments show the effectiveness of STREAM, with stable and significant improvements over different baseline models.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jason Fries, Sen Wu, Alex Ratner, and Christopher Ré. 2017. Swellshark: A generative model for biomedical named entity recognition without labeled data. *ArXiv preprint*, abs/1704.06360.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Athanasios Giannakopoulos, Claudiu Musat, Andreea Hossmann, and Michael Baeriswyl. 2017. Unsupervised aspect term extraction with B-LSTM & CRF using automatically labelled datasets. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 180–188, Copenhagen, Denmark. Association for Computational Linguistics.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification. *ArXiv preprint*, abs/2105.11259.

Zhengbao Jiang, Wei Xu, Jun Araki, and Graham Neubig. 2020. Generalizing natural language analysis through span-relation representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2120–2133, Online. Association for Computational Linguistics.

Martin Krallinger, Obdulia Rabal, Florian Leitner, Miguel Vazquez, David Salgado, Zhiyong Lu, Robert Leaman, Yanan Lu, Donghong Ji, Daniel M Lowe, et al. 2015. The chemdner corpus of chemicals and drugs and its annotation principles. *Journal of cheminformatics*, 7(1):1–17.

Jiacheng Li, Haibo Ding, Jingbo Shang, Julian McAuley, and Zhe Feng. 2021. Weakly supervised named entity tagging with learnable logical rules. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4568–4581, Online. Association for Computational Linguistics.

Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. 2016. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database*, 2016.

Pierre Lison, Jeremy Barnes, Aliaksandr Hubin, and Samia Touileb. 2020. Named entity recognition without labelled data: A weak supervision approach. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1518–1533, Online. Association for Computational Linguistics.

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

Minlong Peng, Xiaoyu Xing, Qi Zhang, Jinlan Fu, and Xuanjing Huang. 2019. Distantly supervised named entity recognition using positive-unlabeled learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2409–2419, Florence, Italy. Association for Computational Linguistics.

Xiang Ren, Ahmed El-Kishky, Chi Wang, Fangbo Tao, Clare R. Voss, and Jiawei Han. 2015. Clustype: Effective entity recognition and typing by relation phrase-based clustering. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pages 995–1004. ACM.

Esteban Safranchik, Shiying Luo, and Stephen H. Bach. 2020. Weakly supervised sequence tagging from noisy rules. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 5570–5578. AAAI Press.

Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R Voss, and Jiawei Han. 2018a. Automated phrase mining from massive text corpora. *IEEE Transactions on Knowledge and Data Engineering*, 30(10):1825–1837.

Jingbo Shang, Liyuan Liu, Xiaotao Gu, Xiang Ren, Teng Ren, and Jiawei Han. 2018b. Learning named entity tagger using domain-specific dictionary. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2054–2064, Brussels, Belgium. Association for Computational Linguistics.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *ArXiv preprint*, abs/2109.01652.

Yunyi Zhang, Jiaming Shen, Jingbo Shang, and Jiawei Han. 2020. Empower entity set expansion via language model probing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8151–8160, Online. Association for Computational Linguistics.

GuoDong Zhou and Jian Su. 2002. Named entity recognition using an HMM-based chunk tagger. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 473–480, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.