Few Clean Instances Help Denoising Distant Supervision

Yufang Liu*[1], Ziyin Huang*[1], Yijun Wang[2],Changzhi Sun[3],
Man Lan[1], Yuanbin Wu[1], Xiaofeng Mou[4] and Ding Wang[4]
[1]School of Computer Science and Technology, East China Normal University
[2]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[3]Bytedance AI Lab, [4]AI Innovation Center, Midea Group
{yfliu.antlp, zyhuang.cs}@gmail.com, ybwu@cs.ecnu.edu.cn

Abstract

Existing distantly supervised relation extractors usually rely on noisy data for both model training and evaluation, which may lead to garbage-in-garbage-out systems. To alleviate the problem, we study whether a small clean dataset could help improve the quality of distantly supervised models. We show that besides getting a more convincing evaluation of models, a small clean dataset also helps us to build more robust denoising models. Specifically, we propose a new criterion for clean instance selection based on influence functions. It collects sample-level evidence for recognizing good instances (which is more informative than loss-level evidence). We also propose a teacher-student mechanism for controlling purity of intermediate results when bootstrapping the clean set. The whole approach is model-agnostic and demonstrates strong performances on both denoising real (NYT) and synthetic noisy datasets.[1]

1 Introduction

Distant supervision was introduced to tackle the lacking training data problem in information extraction tasks (Mintz et al., 2009). By aligning relation triples in knowledge bases (KB) with free texts, it automatically builds labelled sentence instances and easily extends the scale of training set to hundreds of thousands samples. Due to this great scalability, distantly supervised entity relation extractors have been extensively studied in the past decade.

Like other weak signals, the major problem about these automatically generated datasets is label noise: not all aligned sentences carry the same semantic of a KB triple (e.g., not all sentences containing "Obama" and "United States" express a "born in" relation). Some applications (e.g., slot filling of the TAC KBP track (Ji and Grishman, 2011)) could be less affected with the help of *instance bags*, which only needs to seek one correct instance among a bag of aligned sentences. For a more general setting which aims to correctly detect relations on *individual sentences* (Miwa and Bansal, 2016; Sun et al., 2018; Wadden et al., 2019; Wang et al., 2020), however, the noisy labels make both learning and evaluation of models vulnerable: we may draw a flawed conclusion by using a dirty test set for a model learned with a dirty training set.

Many methods have been proposed to reduce noise labels (*denoise*) in distant supervision. For bag-level applications, models often rely on attention scores to either filter bad instances inside a bag (intra-bag attentions, Lin et al. 2016) or filter bags full of noisy instances (inter-bag attentions, Ye and Ling 2019). The dilemma there is that, while we expect attention scores to indicate correct labels, we have to train them to fit noisy labels since ground truth labels are noisy. The same difficulty also exists in recent instance-level denoising methods (Qin et al., 2018a,b) where the reward of denoising an instance is obtained by querying noisy labels. Therefore, not only the extraction models but also denoising models are questionable if only noisy labels are given.

In this paper, we would like to restate the importance of trustful data (*clean dataset*) in building large-scale information extraction systems. Specifically, if a *small* clean dataset ($\approx 10^2$ samples) is available, we ask whether the robustness of both the denoising model and final extraction model could be improved.

We start from training a relation classifier on the clean set and propose a new criterion to select good instances from the dirty set. The main idea is that if a testing instance is correctly labelled by distant supervision, some instances in the clean set should support it, and if we remove those support instances, prediction error of the testing instance

*These authors contributed equally.
[1]Our codes are publicly available at: https://github.com/Airuibadi/IF_DSRE.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2528–2539
October 12–17, 2022.

will increase. Comparing with previous work, the criterion is based on perturbation analyses of classifiers instead of directly using output probabilities (scores) of classifiers. Our tool is *influence function* (IF; Cook and Weisberg 1982; Koh and Liang 2017) which can effectively approximate how a classifier's parameters change when removing a training point.

Next, to incrementally explore the dirty set, we compile our instance selection algorithm into a bootstrapping process: training a classifier on the current clean set, selecting new clean instances using the classifier and retraining the classifier on the updated clean set. The key challenge is how to control purity of those intermediate datasets: one noisy instance may bring more noisy instances. Existing works are either lack of such strategy, or use heuristic thresholds on classifiers or dataset size (Jia et al., 2019). Here, we propose a *teacher-student style* update for learning intermediate classifiers. It gradually controls the distance between the current model and history models by regularizing discrepancy of their predictions.

Our whole system could be deemed as a data preprocessing method. Comparing with in-model denoising method (e.g., attention scores), it outputs a new clean set which can be applied to any information extraction models (*model-agnostic*). We conduct experiments on both real distantly supervised datasets (NYT) and synthetic datasets (built on ACE05). The results demonstrate that besides effectively selecting good instances, the influence-function-based criterion can stratify noisy instances according their difficulties for prediction (or importances for a better extractor). We also find that the teacher-student update especially helps when the proportion of incorrectly labelled instances is large. Finally, when learned with clean sets built by our methods, we are able to achieve competitive extraction performances on *manually* labelled testing set.

2 Preliminary

Distantly Supervised Relation Classification Given an entity pair (e_h, e_t) and a sentence s containing the pair, we consider the task of determining whether the entity pair expresses certain relation $r \in R$, where R is the set of relation types (`None` indicates no relation). Denote $x = (s, e_h, e_t, r)$ to be an *instance*, $y \in \{0, 1\}$ to indicate whether x is positive or negative, and $D = \{(x_i, y_i)\}_{i=1}^{|D|}$ to be a set of labelled instances. For simplicity, we also define $z = (x, y)$.

In the distant supervision setting, instances in D are automatically obtained by aligning plain text and knowledge bases: for a KB triple (e_h, e_t, r), every sentence containing (e_h, e_t) is labelled with r. Obviously, D is a dirty set with both false positives (sentences don't match the semantic of r) and false negatives (sentences expressing relation r while been labelled with `None` due to incompleteness of KB). Here, we focus on false positives (much more serious in current datasets) and don't consider false negatives for its very low quantity. The denoising task is thus to find $D' \subset D$ containing correctly labelled instances (especially, positive instances).

Influence Function (Cook and Weisberg, 1982; Koh and Liang, 2017) provides a way to estimate how individual training instances influence a model. Typically, for a testing instance (x', y'), it efficiently answers the question that if a training instance (x, y) is removed how the model's prediction on (x', y') changes.

Denote $\mathcal{L}(z, \theta)$ to be a convex loss function of z with parameter θ, and $\hat{\theta} \triangleq \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(z_i, \theta)$ to be the optimal model parameter learned on a training set (n is the set size). To study a training instance z's influence on $\hat{\theta}$, influence function considers an ϵ up-weight on z. Define $\hat{\theta}_{\epsilon,z} \triangleq \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(z_i, \theta) + \epsilon \mathcal{L}(z, \theta)$. Therefore, when $\epsilon = -\frac{1}{n}$, $\hat{\theta}_{\epsilon,z}$ is the new model parameter after removing z from the training set.

The key idea of influence function is that, when ϵ is small (or training set size n is large), with the first order Taylor approximation, we can measure the difference between $\hat{\theta}$ and $\hat{\theta}_{\epsilon,z}$ without retraining the model,

$$\hat{\theta}_{\epsilon,z} - \hat{\theta} \approx -\epsilon H_{\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}(z, \hat{\theta}) \triangleq \epsilon \mathcal{I}_{up,params}(z),$$

where $H_\theta = \frac{1}{n} \sum_{i=1}^{n} \nabla^2 \mathcal{L}(z_i, \theta)$ is the Hessian matrix of the original loss function. [2]

We can also get the change of the model's prediction on a testing instance z' by the chain rule,

$$\begin{aligned} &\mathcal{L}(z', \hat{\theta}_{\epsilon,z}) - \mathcal{L}(z', \hat{\theta}) \\ \approx &-\epsilon \nabla_\theta \mathcal{L}(z', \hat{\theta})^T H_{\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}(z, \hat{\theta}) \triangleq \epsilon \mathcal{I}_{up,loss}(z, z'). \end{aligned}$$

We say z *supports* (or is *helpful* to) z' if removing z increases the testing loss of z', that is $\mathcal{S}(z, z') > 0$,

[2] We follow (Koh and Liang, 2017) using a stochastic estimation of H_θ^{-1} when computing influence functions.

where

$$\mathcal{S}(z, z') \triangleq -\frac{1}{n}\mathcal{I}_{up,loss}(z, z'). \quad (1)$$

We will see in the next section that the group of supporting instances is a key part in our denoising algorithms.

3 Utilities of Clean Sets

We study the task of picking out correct instances (D') from a distantly supervised dataset D (the dirty set). As discussed above, it is not easy for the denoising model to either correctly evaluate its results or receive the right learning signals if we only know noisy labels in D. Therefore, departing from previous works, we additionally require a small clean set C ($C \cap D = \emptyset$, $|D| \gg |C|$) which contains trustful annotations of instances (e.g., manually labelled). In our experiments, $|C| = 10$ is enough to bring significant improvement.

We build our denoising model based on a binary classifier $\hat{\theta}$, which aims to recognize truly labelled instances from D.[3] The classifier could be learned on D or C. For example, for an instance z, to test whether it is correctly labelled or not, a broadly applied principle is to query the classifier's confidence on predicting z's label: the lower loss $\mathcal{L}(z, \hat{\theta})$, the more possible z being correctly labelled.

Here, we go one step deeper: besides looking at the loss function, we could first seek high impact training samples on the classifier's drawing of $\mathcal{L}(z, \hat{\theta})$, and then collect evidence from them. For example, the more clean instances support z, the more possible z being correctly labelled. We are going to demonstrate that by probing the black-box classification process, we could build more explainable (yet effective) criteria for selecting instances.

First, from the computation of influence function, we can obtain a training instance's influence on a testing instances (Equation 1). Then, for a instance $z_d \in D$ (as discussed above, we mainly focus on positive $z_d = (x, y)$ where $y = 1$), we have two possible directions to derive a selection criterion.

- **Criterion 1**. We can train a classifier on D. z_d is correctly labelled if it supports $\hat{\theta}$'s prediction on the clean set C. Concretely, define $\mathcal{S}(z_d, \star) \triangleq \frac{1}{|C|}\sum_{z_c \in C}\mathcal{S}(z_d, z_c)$ to be the marginal $\mathcal{S}$ over the testing set,

$$\mathcal{S}(z_d, \star) > 0 \implies z_d \text{ is correct.} \quad (2)$$

- **Criterion 2**. We can also train a classifier on C. In this case, z_d is correctly labelled if $\hat{\theta}$'s prediction on z_d is supported by the instances in C. Define $\mathcal{S}(\diamond, z_d) \triangleq \frac{1}{|C|}\sum_{z_c \in C}\mathcal{S}(z_c, z_d)$ to be the marginal $\mathcal{S}$ over the training set,

$$\mathcal{S}(\diamond, z_d) > 0 \implies z_d \text{ is correct.} \quad (3)$$

Given the limited budget of clean instances C, the two methods are different in their way of using them. When taking C as the testing set (Criterion 1), we emphasize a valid feedback signal for evaluating the denoising model. On the other hand, when taking C as the training set (Criterion 2), we emphasize a clean learning signal for building the denoising model. We would like to discuss more on their pros and cons.[4]

For Criterion 1, as D is usually large enough, we could obtain a sufficiently learned classifier for denoising. More importantly, a large training set makes the estimation of influence function more reliable (Taylor expansion works on small ϵ). However, a good fitting of the dirty set could be a double-edged sword, especially when the proportion of wrongly labelled instances is large: we do get the influence function estimation right but it may not be applicable to our goal of denoising. We can first consider an ideal setting where all instances in D are true. In this case, Equation 2 is trustable since the ideal parameter $\hat{\theta}'$ is trustable, and it encodes the right information for detecting supporting relationship between the training and testing set. However, if a large part of D is false, the classifier $\hat{\theta}$ can diverge from the ideal $\hat{\theta}'$ severely, thus makes Equation 2 no longer true (e.g., a negative z_d could also satisfy the criterion as $\hat{\theta}$ is learned with noise).

Furthermore, we can have the following characterization of $|\mathcal{L}(z_d, \hat{\theta}') - \mathcal{L}(z_d, \hat{\theta})|$ if $L(z, \hat{\theta})$ is in the form of log-likelihood,

$$L(z, \hat{\theta}) = -\log p(y|x, \hat{\theta}) = -\log \frac{\exp\left(\hat{w}_y^\intercal h(x, \hat{\varphi})\right)}{Z}$$

[3] It is also possible to denoise by directly comparing similarities among instances (e.g., using patterns or sentence embeddings). While these methods are important, we mainly focus on classifier-based models whose settings are more analogous to semi-supervised learning or active learning. Comparing with them is beyond the scope of this paper.

[4] Similarly, we can also select the wrongly labelled instances by selecting the lowest influence function scores, we try to flip the labels and add them to the training set, but we find it barely working. The possible reason can be that these instances are positive instances for other entities or relations which adds too much noise for our classifier.

where $\hat{\theta} = [\hat{w}_0, \hat{w}_1, \hat{\varphi}]$, $\hat{w}_0, \hat{w}_1$ are class label embedding, $h(x, \hat{\varphi})$ is a learned representation of x (*encoder*), and Z is the normalizer.

Lemma 1. *Let $z = (x, y) \in D$, $z' = (x, y')$ be a relabelled z, and $\hat{\theta}_{z,z'}$ be the optimal model parameter after replacing z with z'. Denote τ_x to be the smallest singular value of $\nabla_\varphi h(x, \hat{\varphi})$. Then for any $z_d \in D$, up to $o(n^{-1})$, $|\mathcal{L}(z_d, \hat{\theta}_{z,z'}) - \mathcal{L}(z_d, \hat{\theta})|$ is lower bounded by*

$$\frac{c}{n}\left(\|h(x, \hat{\varphi})\| + \tau_x \|\hat{w}_y - \hat{w}_{y'}\|\right),$$

for some constant c. Proof is in Appendix A.

Therefore, if the classifier $\hat{\theta}$ fits well on the dirty set (in the sense of a large $\|\hat{w}_0 - \hat{w}_1\|$), $\mathcal{S}(z_d, z_c)$ calculated with $\hat{\theta}$ could be far away from its value being calculated with a clean training set (i.e., with $\hat{\theta}'$). For the case of multiple updates, since the group version of influence function may not faithfully reflect the change of parameters (Koh et al., 2019), we are not able to obtain similar results with Lemma 1. However, our empirical evaluations will show that performances of Criterion 1 is highly related to the proportion of clean instances in D.

For Criterion 2, comparing with training with dirty D, C contains trustful data, thus the implication relation in Equation 3 is clear after training on C. However, since the clean set is usually small, Criterion 2 takes the risk of under-fitting, which makes the prediction on $z_d \in D$ not sufficiently exploit structures of clean samples in C. Moreover, the estimation of influence function also becomes unstable on small datasets (i.e., ε is larger). In summary, instead of measuring a wrong $\mathcal{S}(z_d, z_c)$ with good accuracy (like Criterion 1), Criterion 2 may struggle with measuring the right $\mathcal{S}(z_c, z_d)$ with poor accuracy.

In the following section, we investigate bootstrapping methods to enlarge C incrementally. We hope that when the number of clean instances becomes larger, we could alleviate both under-fitting and poor estimation of influence function gradually.

4 Bootstrapping the Clean Set

Given a initial small clean set C_0 and a dirty set D_0,[5] our bootstrapping framework incrementally updates a denoising classifier $\hat{\theta}$. At iteration t, we first collect a fixed-size clean set $\tilde{C}$ by sampling from C_t. Second, a denoising classifier $\hat{\theta}$ is trained on the sampled set $\tilde{C}$, from which we can use influence-function-based scores (Equation 1) to evaluate each instance in D_t and choosing new clean instances D^c from D_t. Third, we update C_t and D_t by merging and excluding instances in D^c and retraining the denoising model again. As discussed above, how to control the purity of those intermediate clean sets is important (otherwise, we will face the same challenge as Criterion 1). We propose teacher-student style update for learning intermediate classifiers. It gradually controls the distance between the current model and history models by regularizing discrepancy of their predictions. We summarize the whole process in Algorithm 1. It is worth noting that the output of the bootstrapping process is a new clean set, on which we could build any relation classifier (i.e., *model-agnostic* denoising).

Denoising Classifier Our denoising model is a binary classification model. For each $x = (s, e_h, e_t, r)$, it predicts $y \in \{0, 1\}$. Here, we simply apply a softmax layer on a CNN encoder (the same setting of Lemma 1).[6] Specifically, $h(x, \varphi) = \text{CNN}(\mathbf{s}, \mathbf{p})$, where $\mathbf{s}$ contains embeddings of words in sentence s, and $\mathbf{p}$ contains position embeddings which indicates two entities e_h, e_t in the sentence (Zeng et al., 2014).

Sampling To obtain a fixed-size clean set $\tilde{C}$, we randomly sample instances from C_t with replacement. We keep $|\tilde{C}| = 200$ so that the influence function calculation is more efficient.

Fitting To fit the relation classifier on the sampled set $\tilde{C}$, our objective is to minimize

$$\hat{\theta} = \arg\min_\theta \sum_{z \in \tilde{C}} \mathcal{L}(z, \theta) \tag{4}$$

The parameters $\hat{\theta}$ are applied in calculating IF.

Evaluating After obtaining the parameters $\hat{\theta}$, to evaluate each instance $z_d \in D_t$, we define a score function by Criterion 2 as follows,

$$\mathcal{S}(\diamond, z_d) \triangleq \frac{1}{|\tilde{C}|} \sum_{z_c \in \tilde{C}} \mathcal{S}(z_d, z_c) \tag{5}$$

The score $\mathcal{S}(\diamond, z_d)$ is the average of clean training instances' influence on the test instance z_d.

[5] We use the subscript t to indicate the number of iterations. In some cases, we drop it for simplicity.

[6] The model could be any existing relation model. For simplicity, we select a simple CNN.

Algorithm 1 Bootstrapping Framework

Require: $C_0, D_0, t_{\max}, k$
Ensure: D^r
1: For the student model, initialize θ randomly
2: For the teacher model, initialize $\bar{\theta}$ with θ
3: **for** $t = 0 : t_{\max}$ **do**
4: Sample $\tilde{C}$ from C_t randomly
5: Fit θ on $\tilde{C}$ by Eq.9
6: Fit $\bar{\theta}$ by Eq. 10
7: Evaluate $\mathcal{S}(\diamond, z_d)\ \forall z_d \in D_t$ by Eq. 5
8: Select D^c by Eq. 6 and Eq. 7
9: Update C_t, D_t by Eq. 8
10: $D^r = C_{t_{\max}} \setminus C_0$

Selecting After obtaining the score for each instance $\mathcal{S}(\diamond, z_d)$, we can select the clean instances from D_t according $\mathcal{S}(\diamond, z_d) > 0$ (Criterion 2). In practice, we observe that adding a relaxation factor works better. Formally, we denote it as follows,

$$\tilde{D}_t^c = \{z_d \in D_t | \mathcal{S}(\diamond, z_d) + r > 0\} \quad (6)$$

where r is a positive number. In addition, we adopt a majority voting strategy: we consider not only the current iteration, but also the previous iterations to build the current cleaned set D^c, denoted as:

$$D^c = \left\{ z_d | \sum_{i=0}^{t} \mathbf{1}(z_d \in \tilde{D}_i^c) > k \right\} \quad (7)$$

where $\mathbf{1}(\cdot)$ is the indicator function and k is a hyper-parameter.

Updating Once we have the set D^c, we can update C_t and D_t with simple set operations, denoted as:

$$C_{t+1} = C_t \cup D^c, \quad D_{t+1} = D_t \setminus D^c \quad (8)$$

Teacher-student Mechanism Even though we have used an implicit majority voting to keep selected instances clean, affected by under-fitting and unstable estimation of influence function, error instances will inevitably enter the clean set. Considering our algorithm is based on bootstrapping, errors in previous rounds would have continuous impact on subsequent selection. As mentioned before, the model parameter θ is easily disturbed by wrong-label instances, with the error propagation, the θ would be rotten quickly. To avoid this case, we introduce a teacher-student mechanism (Tarvainen and Valpola, 2017).

Here,we deem θ as the student model, and use another set of model parameters $\bar{\theta}$ as the teacher model, In the fitting step, we add a consistency regularizer to Equation 4:

$$\hat{\theta} = \arg\min_{\theta} L(\tilde{C}, \theta) + \alpha \mathrm{KL}(q(*; \bar{\theta}) || p(*; \theta)) \quad (9)$$

where the q and p are outputs of teacher model and student model respectively, the KL-divergence provides a consistency loss and α is a hyper-parameter. Furthermore, the $\bar{\theta}$ would not be updated in Equation 9, we update it by exponentially moving average as commonly used in teacher-student method:

$$\bar{\theta}_t = \beta \bar{\theta}_{t-1} + (1 - \beta)\hat{\theta}_t \quad (10)$$

Teacher-student mechanism is seen as a regularization term of $\tilde{\theta}$ during fitting, and we neglect this term when calculating the influence function. After $t_{\max}$ times loop, we remove the seed set C_0 from the $C_{\max}$ and obtain our final result $D^r = C_{t_{\max}} \setminus C_0$. Then, we could train any model on D^r.

5 Experiment

5.1 Configurations

NYT The NYT dataset is a widely-used distant supervision benchmark, which is built by Riedel et al. (2010) and rearranged by Jia et al. (2019). The training set is annotated with distant supervision while both development set and test set are manually annotated. For this dataset, We set D to be the NYT training set, and C as to be the NYT development set. [7]

ACE05-N The ACE05-N dataset is a synthetic noisy dataset which adapted from ACE05 (Walker et al., 2006), a commonly used instance-level supervised dataset. We first add the same amount of negative instances (with `None` relation label) as the annotated instances, and then mix additional noisy instances with different ratio, which are flipped `None` instances. Detailed dataset specification could be found in the supplementary.

Settings The settings and implementation details are in Appendix B. We evaluate Precision, Recall, and F1 with micro-averaging in instance-level.

[7] Noting that there is no instance leakage in the following evaluation on development set: we have remove it from our obtained clean set (line 10 of Algorithm 1).

Encoder	Method	Dev Prec.	Dev Rec.	Dev F1	Test Prec.	Test Rec.	Test F1
CNN	RL	42.50	**71.62**	53.34	43.70	72.34	54.49
	Conf	**83.41**	56.03	67.03	58.09	58.09	67.75
	Cr1	81.33	43.94	57.06	73.49	43.62	54.75
	Cr2	76.82	61.54	68.34	**79.71**	60.48	**68.78**
	Cr2TS	76.80	62.10	**68.69**	75.36	**60.52**	67.13
PCNN	ATT	68.09	47.49	55.95	67.31	49.83	57.27
	Conf	82.18	57.23	67.47	**80.15**	58.48	67.62
	Cr1	78.38	47.64	59.26	76.63	52.19	56.85
	Cr2	75.94	**62.87**	68.78	78.71	**61.86**	**69.27**
	Cr2TS	**79.34**	61.14	**69.06**	79.60	59.62	68.17
BiLSTM	ARNOR	78.14*	59.82*	**67.77***	**79.70***	**62.30***	**69.93***
	Conf	80.37	55.82	65.88	79.46	56.43	65.99
	Cr1	**80.73**	55.28	62.06	69.28	54.16	60.79
	Cr2	72.39	**60.67**	66.01	72.04	61.85	66.56
	Cr2TS	77.38	60.00	67.59	74.90	58.65	65.78

Table 1: Comparison of our method and other baselines with different encoders. We denote Conf, Cr1, Cr2, Cr2TS as Confidence, Criterion 1, Criterion 2 and Criterion 2 with teacher-student update style. The code of ARNOR is not accessible now and we find it is hard to reproduce the reported performances.

5.2 Baselines

ATT (Lin et al., 2016) is a classical bag-level denoising method which tunes the attention weight of each instance in bags during training to alleviate the impact of noisy instances.

RL (Qin et al., 2018b) introduces reinforcement learning method to train a instance selector that could tell the noisy instances from the distant supervised training set.

ARNOR (Jia et al., 2019) embeds the relation pattern attention based on recurrent neural network into a bootstrapping framework.

Confidence We implement another baseline for fair comparison, which uses the trained model parameters $\hat{\theta}$ to select instances by confidence each iteration instead of influence function criteria. As a control, it also starts from a initial clean seed set.

5.3 Main Result

Table 1 lists overall performances on NYT dataset with different relation classification models (recall that our approach is model-agnostic). We compare the results of our method with several baselines. From the results, we find that,

- Comparing with prior methods, both Cr2 and Cr2TS achieve better or comparable performance with different encoders, which suggests that our model-agnostic method could effectively prevent RE model from noise data.
- Both Conf and Cr2 use the dev data as a reference, while Cr2 achieves superior performance. Thus, our method is a better strategy which makes use of limited clean set. We credit it to that the influence function could select better instances under the criterion 2.
- The results show that Cr1 performs much worse than Cr2 on both dev and test set. As we mentioned before, Cr1 uses dirty set as training set, leading to unreliable influence.
- The performance of Cr2TS is worse than that of Cr2, which shows the teacher-student mechanism has no advantage on this dataset. We guess it's relative to the noise ratio on dirty set, and further discuss in next section.

6 Analysis and Discussion

Validating influence function. The calculation of influence function is the key step of our method. Here we show the high correlation between the real influence (calculated by leave-one-out retraining) and estimated influence. From the experimental results (see Appendix E), we find that the correlation among high influential instances is 0.79, and 0.65 in all instances. The high correlation validate that influence function is reliable in perform instance perturbation analyses.

Bootstrapping process in detail. In this section, we study the performance change of four selecting strategies during the bootstrapping procedure, as show in Figure 1.

- The performance curves of four strategies are quite similar, which gradually rise to the peak at the beginning and then fall to the line of original noisy data. We think the main reason for this phenomenon is that these strategies add more clean instances into D_l^c in the early epochs, and inevitably select more and more noisy instances in the later training epochs.
- The curve of Cr2 and Cr2TS is higher than Conf, which suggests that the effectiveness of criterion 2. As expected, Cr1 fails in the later period, which is even inferior to training with original noisy data.

The impact of noise ratio. We conduct experiments with different ratio of noise data to verify the denoising ability of our method (Figure 2).

- Even the noise ratio is extreme high (90%), the Cr2 and Cr2TS is still stable. We think that

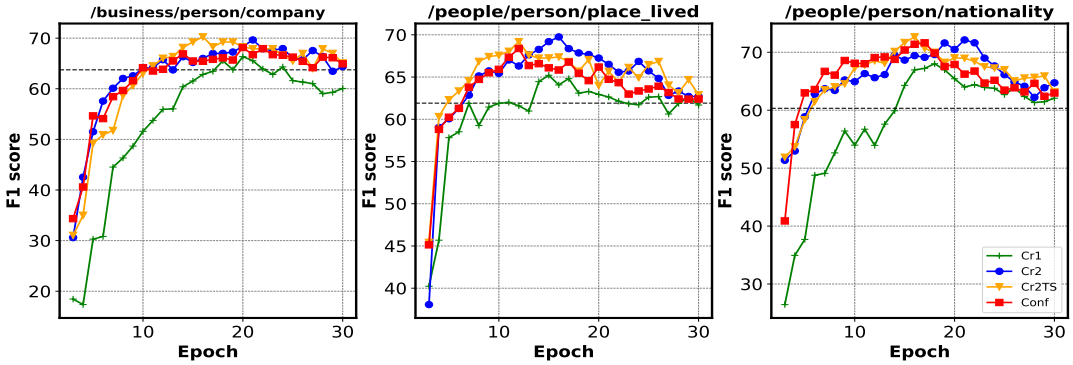

Figure 1: Bootstrapping result. Here we take 3 NYT relations as examples, we present performance change on dev during the bootstrapping. The black dash lines are the performance without denoising.

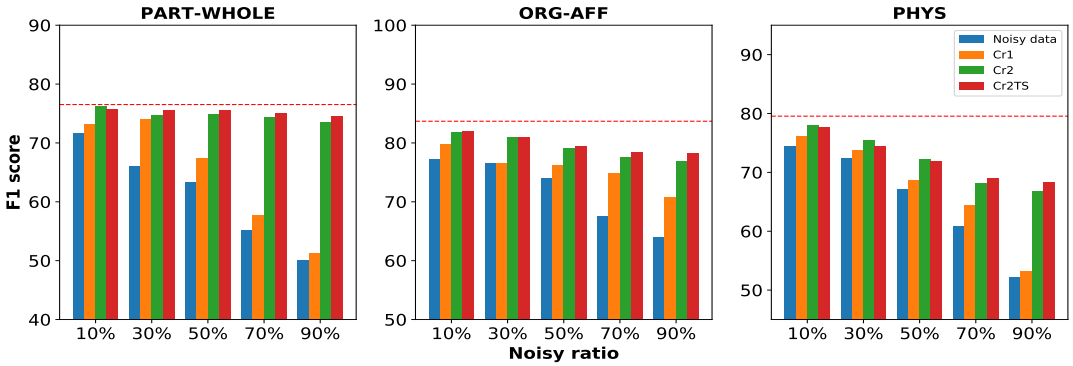

Figure 2: The denoising ability of our method with different noise ratio. Here we take three relation types from ACE05-N as examples. We evaluate their performances with the different ratio of noisy instances that range from 10% to 90% for each relation type. The red dash line indicates the performance without any noisy.

our methods take the most of the clean set to distinguish clean instances from the dirty set, so the damage from the noise instances in dirty set is quite slight.

- The `Cr1` would crash when the noise ratio beyond a certain level, as we analysis before, the higher noise ratio the train set has, the wider gap between estimated influence and the real influence of noisy instances.
- It is worth noting that `Cr2TS` would be better than `Cr2` in the case of the high noise ratio, which shows the effectiveness of teacher-student mechanism. The teacher-student mechanism has the advantage to help lower the lasting impact of misclassified instances in the previous iteration.

The impact of initial clean set size. Our method starts with an initial clean set, so we study the impact of the set size in Table 2.

- In general, the performance goes down with the number of clean instances decreasing. That is reasonable for that the limited clean seed set would suppress the methods to find more true positive instances.
- The performance of `Conf` drops sharply with few initial clean instances (10 instances). We guess that the method only considering confidence of instances is easily trapped into the limited clean set and hard to detect more clean instances.
- Both `Cr2` and `Cr2TS` show better robustness even the the size is extreme small. We believe that the key factor is the influence func-

Methods	10	30	50	ALL
Conf	31.43	34.28	35.89	36.95
Cr1	27.93	28.22	28.63	30.73
Cr2	35.1	36.23	37.42	37.87
Cr2TS	36.52	37.82	38.04	38.69

Table 2: We conduct our experiment on ACE05-N with 50% noise ratio to study the impact of initial clean set size. Note that the F1-score is 27.50 without any selecting strategy. Columns represents the result of using different number of instances as initial clean set. For each relation, we try to use 10, 30, 50 and all dev set as the intial clean set.

Reference:Jeffrey Katzenberg, chief executive of DreamWorks Animation, said...			
	Sentence	**TP/FP**	**Score**
Obvious TP	...Richard C.Notebaert, the cheif excutive of Qwest...	TP	3.46e-3
	...and Bruce Wassertein,the chairman and cheif excutive of Lazard	TP	1.73e-3
Hard instances	...last October, Ray Ozzie, chief technical officer, who joined Microsoft last year ...	TP	3.39e-5
	...Richard C.Noteaert, the company's chief executive, said Qwest spent..	FP	4.91e-5
Potential FP	Eric Foner is the De Witt Clinton professor of history at Columbia University and the author...	TP	-2.36e-3
	As Bruce Wasserstein left St. Regis Hotel in Manhattan on Tuesday afternoon after presenting Lazard's plan ...	FP	-4.37e-3

Table 3: An example of layered phenomenon of instances in the noisy dataset. We group instances by their scores calculated by `Cr2`.

tion, which considers more than confidence and is more practical to extend the scale of clean instances from a small start.

Stratification of instances. Table 3 presents a stratification of instances in the noisy dataset, which is our source of inspiration. There are three layers sorted by the score with Criterion 2.[8] The first layer contains instances with large positive score, which usually have a similar syntactic and semantic structure with the reference instance. These instances are true positive instances, and our method select them in every iteration. The second layer is made up of instances with score around zero. These instances are usually hard to tell whether they are noisy or not. The true positive instances in this layer could be discovered by extending clean set with bootstrapping. The last layer is formed by instances with large negative scores, which are quite different from the reference instance. Some of these instances are indeed noise, while some are still true positive instances but just not be supported by this reference instance. These true positive instances would be supported by other reference instances in clean set which selected by the average score in Criterion 2.

[8]We just take one reference instance as example, rather than the average of all reference instances in criterion 2.

7 Related Work

We focus on distant supervision relation extraction via influence function in this paper. For relation extraction, various neural networks like CNN (Zeng et al., 2014, 2015), RNN (Zhang et al., 2015) and Tree-GRU (He et al., 2018). Distant supervision provides a method to automatically label massive training data (Mintz et al., 2009), meanwhile, bringing excessive wrong label instances, so called noise, which stems the training.

To solve the noisy problem, people first take a multi-instance learning methods (Surdeanu et al., 2012; Lin et al., 2016; Ye and Ling, 2019), which puts instances with same entity pair into bags, to alleviate the impact of noisy instances. Then, to make training process closer to real-world application, people focus on instance-level denoising method. An instance-selector is utilized to pick out trustable instance, which is trained by reinforcement learning (Qin et al., 2018b; Feng et al., 2018) and adversarial learning (Qin et al., 2018a). The bootstrapping framework (Jia et al., 2019; Li et al., 2020) is also utilized to promote the ability of classification model gradually from a small seed. For influence function, which is commonly used in robust statistics (Cook and Weisberg, 1982), Koh and Liang (2017) introduce it in machine learning area. As a technology that is aiming to analyse the every training points' influence on model prediction, influence function is widely applied. Ren et al. (2020) apply influence function on weighting unlabeled data to promote semi-supervised learning. Xu and Kazantsev (2019) utilize influence function to designed an efficient strategy for active learning.

8 Conclusion

In this paper, we propose a model-agnostic denoise method for distant supervision relation extraction. We start from training a relation classifier on the clean set and propose a new criterion to select good instances from the noisy data. We leverage the criterion in a bootstrapping learning to extent the clean set iteratively. Further, we propose a teacher-student to control the update. Our method has a strong performance on NYT dataset and shows robustness under the high noise ratio circumstance or very limited size of initial clean set on the synthetic ACE05-N dataset.

9 Acknowledgments

We thank all anonymous reviewers for their constructive and helpful feedback. The corresponding author is Yuanbin Wu. This research was supported by NSFC (62076097) and a joint research fund from AI Innovation Center, Midea Group.

References

R. Dennis Cook and Sanford Weisberg. 1982. *Residuals and Influence in Regression*. New York: Chapman and Hall.

Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. 2018. Reinforcement learning for relation classification from noisy data. pages 5779–5786.

Zhengqiu He, Wenliang Chen, Zhenghua Li, Meishan Zhang, Wei Zhang, and Min Zhang. 2018. See: Syntax-aware entity embedding for neural relation extraction.

Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1148–1158, Portland, Oregon, USA. Association for Computational Linguistics.

Wei Jia, Dai Dai, Xinyan Xiao, and Hua Wu. 2019. ARNOR: Attention regularization based noise reduction for distant supervision relation classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1399–1408, Florence, Italy. Association for Computational Linguistics.

Pang Wei Koh, Kai-Siang Ang, Hubert H. K. Teo, and Percy Liang. 2019. On the accuracy of influence functions for measuring group effects. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5255–5265.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894. PMLR.

Zhenzhen Li, Jian-Yun Nie, Benyou Wang, Pan Du, Yuhan Zhang, Lixin Zou, and Dongsheng Li. 2020. Meta-learning for neural relation classification with distant supervision. *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*.

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.

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.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, Berlin, Germany. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018a. DSGAN: Generative adversarial training for distant supervision relation extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 496–505, Melbourne, Australia. Association for Computational Linguistics.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018b. Robust distant supervision relation extraction via deep reinforcement learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2137–2147, Melbourne, Australia. Association for Computational Linguistics.

Zhongzheng Ren, Raymond A. Yeh, and Alexander G. Schwing. 2020. Not all unlabeled data are equal: Learning to weight data in semi-supervised learning.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part III*, volume 6323 of *Lecture Notes in Computer Science*, pages 148–163. Springer.

Changzhi Sun, Yuanbin Wu, Man Lan, Shiliang Sun, Wenting Wang, Kuang-Chih Lee, and Kewen Wu. 2018. Extracting entities and relations with joint minimum risk training. In *Proceedings of the 2018*

Conference on Empirical Methods in Natural Language Processing, pages 2256–2265, Brussels, Belgium. Association for Computational Linguistics.

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.

Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. 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 1195–1204.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. In *In Linguistic Data Consortium*.

Yijun Wang, Changzhi Sun, Yuanbin Wu, Junchi Yan, Peng Gao, and Guotong Xie. 2020. Pre-training entity relation encoder with intra-span and inter-span information. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1692–1705, Online. Association for Computational Linguistics.

Minjie Xu and Gary Kazantsev. 2019. Understanding goal-oriented active learning via influence functions.

Zhi-Xiu Ye and Zhen-Hua Ling. 2019. Distant supervision relation extraction with intra-bag and inter-bag attentions. 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 2810–2819, Minneapolis, Minnesota. 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.

Shu Zhang, Dequan Zheng, Xinchen Hu, and Ming Yang. 2015. Bidirectional long short-term memory networks for relation classification. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, pages 73–78, Shanghai, China.

Supplementary Materials for
Few Clean Instances Help Denoising Distant Supervision

A Proof of Lemma 1

Proof. Following the same first order Taylor approximation as influence function, up to $o(n^{-1})$, we can get the update of parameters after replacing z with z' (see Equation 3 of (Koh and Liang, 2017)),

$$\begin{aligned}\hat{\theta}_{z,z'} - \hat{\theta} &= n^{-1}\left(\mathcal{I}_{up,params}(z) - \mathcal{I}_{up,params}(z')\right) \\ &= n^{-1} H_{\hat{\theta}}^{-1}\left(\nabla_\theta \mathcal{L}(z,\hat{\theta}) - \nabla_\theta \mathcal{L}(z',\hat{\theta})\right).\end{aligned}$$

Since the set D is finite, we denote

$$c_1 \triangleq \arg\min_{(z,z')} \frac{|\mathcal{L}(z,\hat{\theta}_{z,z'}) - \mathcal{L}(z,\hat{\theta})|}{\|\hat{\theta}_{z,z'} - \hat{\theta}\|},$$

then

$$\begin{aligned}&|L(z_d,\hat{\theta}_{z,z'}) - L(z_d,\hat{\theta})| \geq c_1 \|\hat{\theta}_{z,z'} - \hat{\theta}\| \\ =&c_1 \|H_{\hat{\theta}}^{-1}\left(\nabla_\theta \mathcal{L}(z,\hat{\theta}) - \nabla_\theta \mathcal{L}(z',\hat{\theta})\right)\| \\ \geq&c_1 (n\sigma)^{-1} \|\nabla_\theta \mathcal{L}(z,\hat{\theta}) - \nabla_\theta \mathcal{L}(z',\hat{\theta})\|,\end{aligned}$$

where σ is the maximum singular value of the Hessian. Regarding the log-likelihood loss, $\nabla_\theta \mathcal{L}(z,\hat{\theta}) = \nabla_\theta \log Z - \nabla_\theta \hat{w}_y^\intercal h(x,\hat{\varphi})$, and the transpose of the second term is

$$\begin{matrix} & & w_y & & \varphi \\ \big(& \mathbf{0}, & h^\intercal(x,\hat{\varphi}), & \hat{w}_y^\intercal \nabla_\varphi h(x,\hat{\varphi}) & \big). \end{matrix}$$

Hence,

$$\begin{aligned}&\|\nabla_\theta \mathcal{L}(z,\hat{\theta}) - \nabla_\theta \mathcal{L}(z',\hat{\theta})\| \\ =&\sqrt{2\|h(x,\varphi)\|^2 + \|\nabla_\varphi h(x,\varphi)^\intercal (w_y - w_{y'})\|^2} \\ \geq&\|h(x,\varphi)\| + \frac{\tau_x}{\sqrt{2}}\|w_y - w_{y'}\|.\end{aligned}$$

Let $c = c_1(\sigma\sqrt{2})^{-1}$, we get the lower bound. □

B Implementation details

For basic CNN model, the window size of the convolution layer is set to 3 and the number of the filter is set to 230. In bootstrapping procedure, the position embedding dimension of CNN is set to 1 and the word embedding is initialized with 100 dimensional pre-trained glove embedding (Pennington et al., 2014),[9] which is for making IF focuses more on semantic information. In training procedure, for fair comparison, the position embedding dimension of all models is set to 5, the word embedding dimension is set to 100 with random initialization and an entity type embedding. And for the PCNN model we have the same hype-parameters with Zeng et al. (2015). For majority vote,we set $k = 3$. In selection step of bootstrap, we at most select $n = \frac{1}{10}|D_t|$ instances. For teacher-student, the α is set to 1 and β is set to 0.9. To avoid the model just memorizes the entity pairs, we mask the entity words in both bootstrapping and training.

NYT	Training	Dev	Test
# Sentence	233038	1596	1596
# Instance	367596	4567	4484
# Positive instances	106653	975	1050

Table 4: Statistics on NYT dataset.

NYT	Training	Dev	Test
# /people/person/place_lived	7197	198	185
# /location/location/contains	51766	479	611
# /people/person/nationality	8079	117	91
# /business/person/company	5595	105	113
# /people/person/children	506	6	11
# /people/dec.../place_of_death	1936	8	14
# /location/country/capital	7690	14	15
# /business/company/founders	800	10	6
# /people/person/place_of_birth	3173	13	15
# /location/nei.../nei..._of	5553	6	7

Table 5: 10 relations on dev set. Our methods take the dev set as clean set to denoise.

C Detailed statistics on NYT dataset

Overall statistics on NYT datset are shown in Table 4. And the detailed statistics of each relations on dev set are in Table 5.

D Synthetic dataset ACE05-N

Table 6 shows the statistics on ACE05. Next we show how to construct our synthetic dataset

[9] Download from https://nlp.stanford.edu/projects/glove/.

ACE05-N	Training	Dev	Test
# GEN-AFF	290	73	55
# ORG-AFF	857	204	203
# PER-SOC	279	57	45
# PHYS	549	164	123
# PART-WHOLE	393	81	86
# ART	275	52	85
# NA	116572	27597	24363

Table 6: The statistics on ACE05.

NYT	Training	Dev	Test
# GEN-AFF	580	146	110
# ORG-AFF	1714	408	406
# PER-SOC	556	104	90
# PHYS	1098	328	246
# PART-WHOLE	786	162	172
# ART	550	104	170

Table 7: The statistics of ACE05 after adding NA.

ACE05-N, which takes two steps: adding NA and adding noise. The entity pair in a sentence that does not express any positive relation would be considered as "NA". The NA instances play two roles in our synthetic dataset: the negative instances and noisy.

Adding negative instances In DSRE, the noise come from the wrong-labelled negative instances. So the true negative instances are necessary for evaluating the denoise ability. In our synthetic dataset, we reconstruct the training, dev and test of each relation by adding the NA. The size of NA is same as the original size of training, dev and test. After that, the dataset would be changed to Table 7.

Adding noisy An instance that don't express relation r but labelled with r is a noise instance to the relation r. So the intentionally made noise is relabelling a NA instance to a positive label. In the experiment, we manually put a certain number of noise instance to poison the dataset. If we poison a train set with 50% noise, we mean that after poisoning, the noise ratio of this train set is 50%. For example, if the train set of *GEN-AFF* in Table 7 is poisoned with 50% noise, it would be put with 580 noise instances.

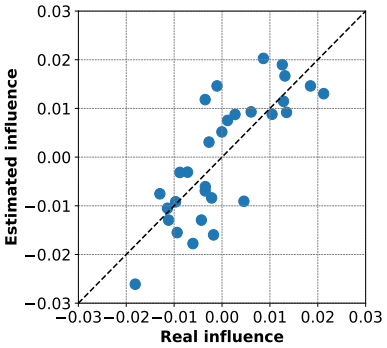

Figure 3: The correlation between estimated change of loss and real change of loss. We use a training set with 500 instances included two relation types and arbitrarily pick four instances as testing instances to validate computation of influence function. The picture shows 40 most influential points with their real difference in loss (obtained by 500 steps leave-one-out retraining).

E Validating influence function

The calculation of influence function is the key step of our method. Here we show the high correlation between the real influence (calculated by leave-one-out retraining) and estimated influence. From the experimental results (Figure 3), we find that the correlation among high influential instances is 0.79, and 0.65 in all instances. The high correlation validate that influence function is reliable in perform instance perturbation analyses.