Multilevel Heuristics for Rationale-Based Entity Relation Classification in Sentences

Shiou Tian Hsu, Mandar Chaudhary and Nagiza F. Samatova
North Carolina State University
shsu3@ncsu.edu, mschaudh@ncsu.edu, samatova@csc.ncsu.edu

Abstract

Rationale-based models provide a unique way to provide justifiable results for relation classification models by identifying rationales (key words and phrases that a person can use to justify the relation in the sentence) during the process. However, existing generative networks used to extract rationales come with a trade-off between extracting diversified rationales and achieving good classification results. In this paper, we propose a multilevel heuristic approach to regulate rationale extraction to avoid extracting monotonous rationales without compromising classification performance. In our model, rationale selection is regularized by a semi-supervised process and features from different levels: word, syntax, sentence, and corpus. We evaluate our approach on the SemEval 2010 dataset that includes 19 relation classes and the quality of extracted rationales with our manually-labeled rationales. Experiments show a significant improvement in classification performance and a 20% gain in rationale interpretability compared to state-of-the-art approaches.

1 Introduction

The goal of sentence-level entity relation classification is to infer how two target entities are semantically associated in a sentence. It is a core NLP function that supports many high level tasks such as information extraction and knowledge graph population (Hendrickx et al., 2009; Niu et al., 2012). Recent advances in generative models facilitate improved classification performance along with justifiable results which improves model interpretability. A generative model will first extract the most representative fragments in the sentence that expresses the relation first, and augments the classifier with the rationales (Hsu et al., 2018) . These representative fragments are called rationales as per (Zhang et al., 2016; Lei et al., 2016; Hsu et al., 2018), and can be used to justify the results. We illustrate an example of rationale-based models using an example sentence which expresses a Instrument-Agency(e2,e1) relationship between the given target entities e_1 = "attacker" and e_2 = "instrument":

Rationale based models: *She had struggled violently with her [attacker]e_1, who* ***killed*** *her* ***with*** *a blunt [instrument]e_2.*
(words marked in bold and the entities are the major sources to infer the relation)

The most commonly used structure in generative models consists of two components: a Generator and a Discriminator. The Generator extracts rationales from an input sentence in an unsupervised fashion; the Discriminator, which is supervised, predicts the relation class utilizing the rationales. However, training a good Generator can be essentially hard due to the mode collapse problem (Chen et al., 2016; Che et al., 2017; Duhyeon and Hyunjung, 2018). Mode collapse is a commonly observed paradox where the Generator attempts to extract varied rationales, but converges to select monotonous rationales because the Discriminator failed to model diversified rationales. A collapsed Generator is not capable of providing

Proceedings of the 27th International Conference on Computational Linguistics, pages 1145–1155
Santa Fe, New Mexico, USA, August 20-26, 2018.

meaningful rationales since it extracts the same rationales repeatedly regardless of the context. However, as shown in Figure 1, a collapsed Generator does not always lead to poor classification performance when compared to a finely converged Generator.

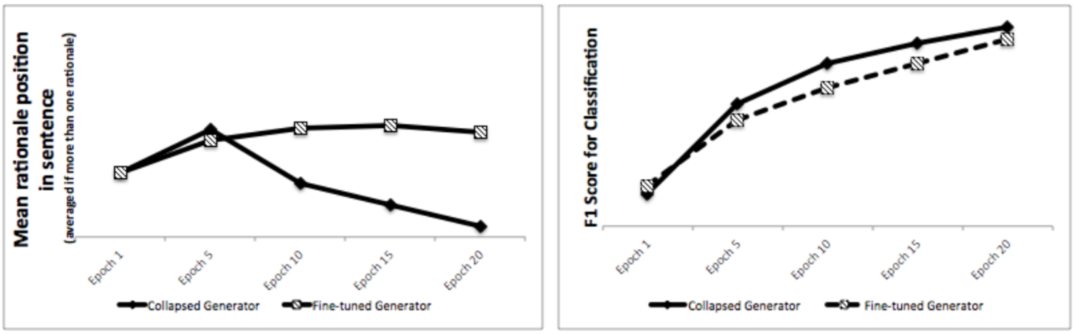

Figure 1: A Generator can converge at collapsed rationales but still lead to good classification results. Figure on the left shows the rationale distribution of a collapsed and a well-converged Generator that uses a recurrent neural net . The well-converged Generator evenly selects rationales from a sentence, while the collapsed Generator selects rationales close to the beginning of the sentence. This is because the best scenario for Discriminator is that some rationales are repetitively selected and optimized in every training iteration, which is difficult for the Generator. Eventually, the Generator learns to exploit position features since it is less diversified than words. Figure on the right shows that a collapsed Generator does not necessarily deteriorate the classification performance.

One of the major causes of mode collapse is the lack of regularizing terms to smooth out the Generator(Chen et al., 2016; Arjovsky and Bottou, 2017; Salimans et al., 2016). In this paper, we aim to improve the rationale-based models for relation classification by introducing semi-supervised capacity and addition regularizing force to the Generator to prevent mode collapsing. In our Generator, we develop a multilevel heuristic to replace the context-focused recurrent neural net used by existing approach(Hsu et al., 2018). Our Generator jointly considers features from different levels of the dataset such as contextual words, target entities, sentence syntax features, rationale-relation closeness, and global word distribution in corpus. The details of our improved Generator are shown in Figure 2.

Our Generator jointly considers several aspects such as contextual words, target entities, word distribution in corpus, sentence grammatical features, and selected rationale-relation closeness to regularize rationale selection.

We evaluate our model on the SemEval 2010 Task 8 dataset which includes 19 relation classes. Specifically, we use F1 score to measure the quality of extracted rationales and the classification performance. Given the absence of ground-truth of rationales in the dataset, we manually label the test set and make it publicly available[1]. We empirically demonstrate an improvement in interpretability of the rationales by 20% compared to Hsu et al. (2018), and also an improvement in the F1 score of 90.7 compared to 89.5 in the state-of-the-art model. We summarize the contributions of our approach as follows:

- We propose a multilevel heuristics approach to avoid mode collapse in rationale-based relation classification models.
- We empirically demonstrate our model produces improved relation classification performance and more interpretable rationales compared to other models
- We provide a labeled set of rationales that can be used for evaluation in future rationale research.

2 Related Work

Research for improving generative neural networks have gained much interest in the research community due to the effectiveness and versatility of these models. These improvements can be divided into two categories: architecture-oriented and divergence-oriented.

[1] `https://github.ncsu.edu/shsu3/rationale-improvement`

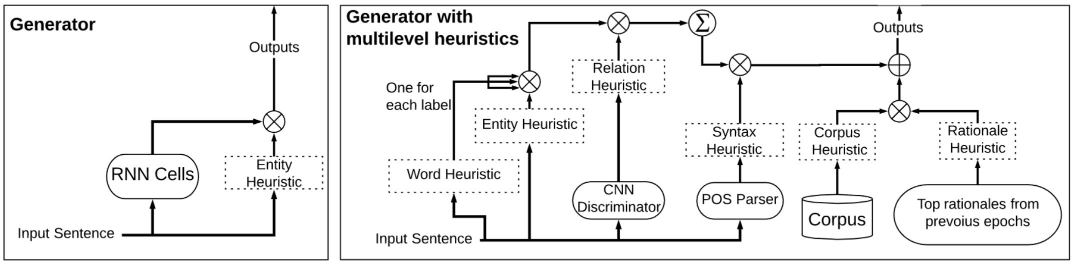

Figure 2: Figure on the left and the right shows the Generator in Hsu et al. (2018) and our proposed multilevel heuristics Generator respectively. The Word heuristic measures the relative closeness of each word in the sentence to every relation class. The Entity heuristic uses the target entities to regulate the Word heuristic. The Relation heuristic contains predictions of the relation from a standalone convolutional neural net model and is also used used to regulate the Word heuristic. The Word heuristic is then summed and weighted by the Syntax heuristic, which considers both, the word distance to entities, and Part-of-Speech labels. The Corpus heuristic is obtained by word-relation distribution in the corpus. The Rationale heuristic is the semi-supervised component which is based on rationales from the previous training iteration.

The Generative Adversarial Network (GAN) (Goodfellow et al., 2014) is the one of most representative model in the architecture-oriented group. Goodfellow et al. (2014) proposed a modified generative model using an adversarial minimax-game formulation. The objective of GAN is to jointly train a Generator and Discriminator, where the loss for Discriminator and the reward for Generator comes from failed attempts to determine ground-truths from the emulated results created by Generator. However, the adversary process in GAN is the major source of mode collapse, hence further extensions of GAN aimed towards addressing this issue. For example, InfoGan (Chen et al., 2016) proposed to maximize the mutual information between a subset of the noise variables with the observations to produce disentangled unsupervised representations. Salimans et al. (2016) proposed to use semi-supervised training and label-smoothing to help GAN converge to Nash Equilibrium between the Generator and the Discriminator. Duhyeon and Hyunjung (2018) proposed RFGAN to implicitly regularize the discriminator using representative features extracted from the data distribution. In VEEGAN (Srivastava et al., 2017), the authors suggested to jointly train an extra Reconstructor with the Generator which is an approximate reverse action of the Generator to encode noise.

Another group of methods have aimed to reduce mode collapse in generative models by modifying the joint loss functions used between the Generator and Discriminator. In Arjovsky et al. (2017), the authors theoretically showed that the KullbackLeibler (KL) divergence often used in GAN was one of the source of instability. Based on this discovery, Arjovsky and Bottou (2017) proposed Wasserstein distance to measure the similarity between the fake and real data to make stabler Generator. Gulrajani et al. (2017) introduce the gradient penalty for regularizing the divergence as an alternative to gradient clipping used in Arjovsky and Bottou (2017) to capture stronger movements in gradients . In DRAGAN (Kodali et al., 2017), they proposed to regulate sharp gradients in the Discriminator, which often happens at some undesired *local equilibria* between Generator and Discriminator. Roth et al. (2017) proposed another type of regularization that breaks down the divergence into different Discriminator output to achieve stability.

We position our work closer to the first category since we can view the multilevel heuristics used in our model close to a combination of models in the first category. The Word and Corpus heuristic are derived from data distribution similar to RFGAN, and the Relation and Rationale heuristic are substantially close to the semi-supervised learning techniques used in Salimans et al. (2016). However, our work is not exclusive to the second category as the Word heuristic included idea close to Roth et al. (2017) where the divergence in the Word heuristic maps to label level in Discriminator instead of treating the Discriminator as a whole.

On a final note, we would like to mention that this work shares some resemblance with (Giuliano et al., 2006) and can be potentially considered as its extension. In Giuliano et al. (2006), the authors utilize several shallow linguistic information for relation classification, such as word-relation frequency or POS to avoid dependent syntactic information. In this work, the fundamental difference is that all token-based computations are replaced by embeddings which are more generalizable when considering words of semantic relatedness. Additionally, the training phase is carried out by the adversarial process.

3 Rationale Generation Framework

We describe the framework for rationale-based approach in the following. Consider an input sentence $\boldsymbol{x}$ = $\{x_t\}_{t=1}^{T}$, where e_1 and e_2 are given target entities of interest and e_1 and $e_2 \subset \{x_1, x_2, ..., x_T\}$. The goal of relation classification is thus to use ($\boldsymbol{x}$,e_1,e_2) to predict the relationship y between e_1 and e_2. We call this prediction process as encoding and denote it as $\boldsymbol{enc}(\boldsymbol{x}$,$e_1$,$e_2)$.

Following Hsu et al. (2018), we consider this rationale based model as an adversarial problem the goal of the Generator is to generate rationales $\boldsymbol{r}$ that $\boldsymbol{enc}(\boldsymbol{r}$,$\boldsymbol{x}$,$e_1$,$e_2)$ can outperform all other sets of rationales in $\boldsymbol{enc}(\tilde{\boldsymbol{r}}$,$\boldsymbol{x}$,$e_1$,$e_2)$. In Hsu et al. (2018), they split the Generator into two steps: the Generator and the Selector. The Generator first generates candidates $\boldsymbol{gen}(\boldsymbol{x}$,$e_1$,$e_2)$, which samples $\boldsymbol{c}$ = $\{c_t\}_{t=1}^{T}$ from the input where $c_t \in [0, 1]$ and it represents whether x_t should be considered as a rationale ; the Selector then samples rationales $\boldsymbol{r}$ from $\boldsymbol{c}$ and is represented as $\boldsymbol{sel}(\boldsymbol{c}$,$e_1$,$e_2)$. We illustrate the framework in Figure 3.

A difference against other generative works is worth noting: in Lei et al. (2016), rationales are defined as a sequence of words in a customer review that are directly related to different rating aspects. This is because in Lei et al. (2016), the assumption is that the reviews contain convoluted sentiments, and the goal of a rationale is to disentangle the sentiments. In effect, the goal of $\boldsymbol{r}$ in Lei et al. (2016) is to serve as a proxy of $\boldsymbol{x}$ that makes $\boldsymbol{enc}(\boldsymbol{r})$ approximate to $\boldsymbol{enc}(\boldsymbol{x})$. while our goal is to find $\boldsymbol{r}$ that outperform all possible short enumerations of $\boldsymbol{x}$ in $\boldsymbol{enc}(\boldsymbol{r}$,$\boldsymbol{x}$,$e_1$,$e_2)$.

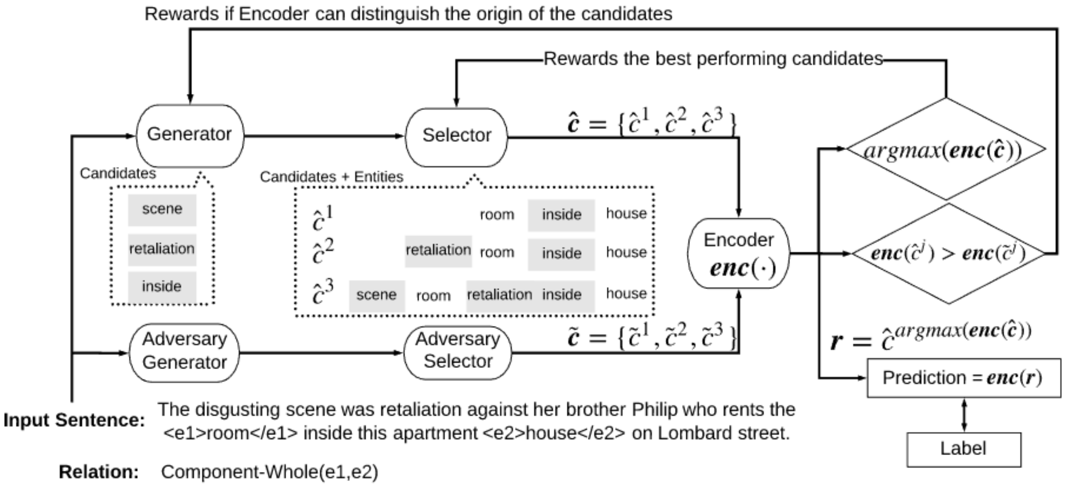
Rewards if Encoder can distinguish the origin of the candidates
Rewards the best performing candidates
Generator
Selector
$\hat{\boldsymbol{c}} = \{\hat{c}^1, \hat{c}^2, \hat{c}^3\}$
$argmax(\boldsymbol{enc}(\hat{\boldsymbol{c}}))$
Candidates
scene
retaliation
inside
Candidates + Entities
$\hat{c}^1$ room inside house
$\hat{c}^2$ retaliation room inside house
$\hat{c}^3$ scene room retaliation inside house
Encoder
$\boldsymbol{enc}(\cdot)$
$\boldsymbol{enc}(\hat{c}^j) > \boldsymbol{enc}(\tilde{c}^j)$
Adversary Generator
Adversary Selector
$\tilde{\boldsymbol{c}} = \{\tilde{c}^1, \tilde{c}^2, \tilde{c}^3\}$
$\boldsymbol{r} = \hat{c}^{argmax(\boldsymbol{enc}(\hat{\boldsymbol{c}}))}$
Prediction = $\boldsymbol{enc}(\boldsymbol{r})$
Label
Input Sentence: The disgusting scene was retaliation against her brother Philip who rents the <e1>room</e1> inside this apartment <e2>house</e2> on Lombard street.
Relation: Component-Whole(e1,e2)

Figure 3: The model structure from Hsu et al. (2018) with a sample input sentence and the given entities. The Generator selects candidate rationales, and the Selector enumerates all possible combinations of candidates with entities and selects one ‘best performing candidate set’. An adversary Generator and Selector carry out the same process using a randomized approach. Candidate sets are then transformed into a vector representation by the Encoder and are evaluated against the ground-truth. Rewards are fed back to the Generator if the Encoder is able to identify candidate sets that are generated randomly, while the Selector is rewarded if the best performing rationale candidate set outperforms the one in the adversary. The best performing candidate set from Selector is finally considered as the extracted rationales, and then used in the Encoder to predict entity relation.

4 Model

In this section, we describe each component of the framework. We begin by explaining the Encoder to formalize the goal of relation classification.

4.1 Encoder

Given an input training tuple ($\boldsymbol{x}$,e_1,e_2,y), where $\boldsymbol{x}$ = $\{x_t\}_{t=1}^{T}$, and y is the one-hot L dimensional vector representing the relation class. The goal of the Encoder $\boldsymbol{enc}$($\boldsymbol{r}$,$\boldsymbol{x}$,e_1,e_2) is to produce a probability distribution $\hat{y}$ that approximates to y as formulated as follows (N is the size of the training set):

$$loss = \sum_{i=1}^{|N|} -y_i \cdot log(\hat{y}_i) \tag{1}$$

The Encoder produces $\hat{y}$ by encoding ($\boldsymbol{r}$,$\boldsymbol{x}$,e_1,e_2) as a vector representation and then projects it to the y space. This vector representation can further break down to a sentence vector V_x and a rationale vector V_r. We produce V_x through CNN using the same model as Kim (2014) which encodes the sentence through a continuous window approach, where K are the set of sizes for the windows. To produce V_r, we implement a RNN model using GRU cells that reads the rationales and target entities. Since RNN is order sensitive, the rationales and the entities are sorted based on their position in the sentence before applying the RNN model.

$$\begin{aligned} V_x &= CNN(\boldsymbol{x}) & \hat{y} &= \boldsymbol{enc}(\boldsymbol{r}, \boldsymbol{x}, e_1, e_2) \\ V_r &= RNN([\boldsymbol{r}, e_1, e_2]) & &= softmax(W_{enc} \cdot [V_r, V_x, e_1, e_2] + b_{enc}) \end{aligned} \tag{2}$$

4.2 Generator

The goal of the Generator $\boldsymbol{gen}(\boldsymbol{x})$ is to derive a set of binary variables $\boldsymbol{c} = \{c_t\}_{t=1}^{T}$ by sampling from probability scores $\boldsymbol{s} = \{s_t\}_{s=1}^{T}$, where $c_t \in [0, 1]$ indicates whether x_t is chosen as a candidate rationale and s_t is computed by the multilevel heuristics. We represent it as $\boldsymbol{c} \sim \boldsymbol{gen}(\boldsymbol{x})$.

In contrast to Hsu et al. (2018) which uses an RNN model for Generator, we adopted a set of logistic regression models with several feed-forward networks to facilitate the Generator. The reason being, an RNN model can be difficult to train and might be susceptible to mode collapse if not carefully tuned. In the Generator, the number of logistic regression models is equal to L which is the number of classes in y. Each word in the sentence is first moderated by the Entity heuristic he, which is obtained from the target entities and fed through the logistic regression models to obtain the Word heuristic $hw = \{\{hw_t^l\}_{l=1}^{L}\}_{t=1}^{T}$. Each hw_t^l can be considered as a probability of x_t being chosen as a rationale if the given sentence is of relation class l. The second step is to compute the Relation heuristic $hr = \{hr^l\}_{l=1}^{L}$ and the Syntax heuristic hs_t that are applied to the Word heuristic. The Relation heuristic initializes the relation with a random guess using the whole sentence. The heuristic is then trained through a CNN model which similar to the CNN model we used to compute Vx. The Syntax heuristic hs_t is obtained from a feed forward neural network that takes as input (1) word distance to target entities, and (2) Part of Speech (POS) tags. The output, s_t^{local}, is obtained by combining the Word, Entity, Syntax and Relation heuristics, and it represents local sentence information score.

The final score s_t is then computed by applying the Corpus and Rationale heuristics to s_t^{local}. The Corpus heuristic $hc = \{hc^l\}_{l=1}^{L}$ contains binary values computed by sorting and selecting the frequent K words in each relation class in the training set. The Rationale heuristic $hra = \{hra^l\}_{l=1}^{L}$ also contains binary values which are based on taking the top portion of the Word heuristic for each relation class, but the Word heuristic considered here is not moderated by entities when computing the Rationale heuristic. The Corpus and Rationale heuristics capture the association between the words and relations are in the dataset. To leverage this information, we construct a global heuristic, s_t^{global}, by computing an element-wise multiplication of hc and hra. Note that s_t^{global} can be seen as a self-learning resource for semi-supervised learning because the Rationale heuristic is based on the results from previous iterations.

Accordingly, s_t^{global} will not be used in testing and will only be included after several training iterations have completed. Finally, before applying s_t^{global} to the final score s_t we apply a dynamically computed weight factor CR to $s_t^{local}$and s_t^{global}. We formulate all the heuristics in the following equation,

$$
\begin{aligned}
&he = sigmoid(W_{he} \cdot [e1, e2] + b_he) && s_t^{local} = hw_t \cdot \mathcal{T}(hr) \cdot hs \\
&hw_t = sigmoid(W_{hw} \cdot (x_t \odot he) + b_he) && \quad (\mathcal{T}\text{stands for transpose function}) \\
&hr = CNN(\boldsymbol{x}) && s_t^{global} = hc \odot hra \\
&hs_t = sigmoid(W_{hs} \cdot [POS_t, position_t]) && s_t = s_t \cdot (1 - CR) + s_t^{global} CR \\
&CR = sigomid(W_{CR} \cdot hr' + b_{CR}) &&
\end{aligned}
\tag{3}
$$

In the most simplistic Generator, the probability that x_t is chosen is conditionally independent of all other $\boldsymbol{x}$. In other words, we can sample candidate $\boldsymbol{c}$ from a uniform distribution using the probability scores, $\boldsymbol{s}$. However, we observe that the target rationales in the dataset often consist of few words - for example "*room* ***inside*** *apartment*". Therefore, we added the following Equation (4) to limit the number of rationales selected, $\boldsymbol{c}$, to be at most J.

$$
\begin{aligned}
c_t &= \begin{cases} 1, sort(\{(s_t > rand(0,1)) \cdot s_t\}_{t=1}^{T})[1:J] \\ 0, otherwise \end{cases} \\
\boldsymbol{c} &= \{c_t\}_{t=1}^{T}
\end{aligned}
\tag{4}
$$

4.3 Selector

The last component of the model is the Selector $\boldsymbol{sel}(\boldsymbol{c})$, where the goal of the Selector is to decide the number of rationales best suited for the prediction. The Selector is facilitated by a simple feed-forward network that outputs a number from $1 \sim J$ by scoring rationale sets of length ranging from $1 \sim J$ based on the top ranked $s_t * c_t$. During training, all rationale sets will be forwarded to the Encoder, and the training label for Selector will be the size of the set that obtains the least training loss in Encoder. This Selector is identical to the one in Hsu et al. (2018) and is defined as follows:

$$
\begin{aligned}
&\hat{c}_t^j = \begin{cases} 1, sort(\{c_t \cdot s_t\}_{t=1}^{T})[1:j], j \in [1:J] \\ 0, otherwise \end{cases} && scores^j = softmax(\{score(\hat{\boldsymbol{c}}^{\boldsymbol{j}})\}_{j=1}^{J})^j \\
& && sc\hat{o}res^j = \begin{cases} 1 & , j = argmax(scores) \\ 0 & , otherwise \end{cases} \\
&\hat{\boldsymbol{c}}^{\boldsymbol{j}} = \{\hat{c}_t^j\}_{t=1}^{T}, \text{where } \hat{c}_t^j = 1 && \boldsymbol{r} = \{x_t\}, \text{where } \hat{c}_t^j = 1,\ j = argmax(scores) \\
&score(\hat{\boldsymbol{c}}^{\boldsymbol{j}}) = W_c \cdot \boldsymbol{enc}(\hat{\boldsymbol{c}}^{\boldsymbol{j}}, \boldsymbol{x}, e1, e2) &&
\end{aligned}
\tag{5}
$$

4.4 Joint Objective and Adversarial Training

We formulate the joint loss function in the following to bind the components.

As described earlier, the goal of the Generator and the Selector is not to emulate $\boldsymbol{x}$ with $\boldsymbol{r}$ due to the nature of this research. In effect, the Generator and the Selector are not competing against the Encoder, but instead with an adversary Generator, for which we use a randomized approach. The adversary Generator will sample $\tilde{\boldsymbol{c}}$ randomly from $\boldsymbol{x}$. The adversary rationales $\tilde{\boldsymbol{r}}$ will then be selected using $\tilde{\boldsymbol{c}}$ as per Equation (4). $\tilde{\boldsymbol{r}}$ will be passed to the Encoder to generate a projection $\tilde{y}$ onto the target dimension. We compare $\tilde{y}$ with $\hat{y}$ in terms of their similarity to y and is denoted by $\mathcal{D} \in \{0, 1\}$. $\mathcal{D} = 1$ when $\hat{y}$ is closer to y compared to $\tilde{y}$, and is noted as a 'model success case', meanwhile $1 - \mathcal{D} = 1$ for the opposite situation and is noted as an 'adversary success case'. The objective for the Generator and Encoder reacts differently to the two different cases, where the Encoder optimize whichever rationales that performs better, and the Generator rewards the selected rationales in model success case and penalize in the other case. We further split $\hat{y}$, $\tilde{y}$ and $\mathcal{D}$ into $\hat{y}^j$, $\tilde{y}^j$, $\mathcal{D}^j$ and $j \in [1:J]$, where $\mathcal{D}^j$ represents that $\hat{y}^j$ outperforms $\tilde{y}^j$ when using j rationales. Finally, we introduce a penalizing factor P^j when $j \neq argmax(scores)$ to penalize the gradients from poor performing cases. We summarize as follows:

$$
\begin{aligned}
f(s_t^{'jl}) &= -log(s_t^{'jl}) \cdot c_t^j \cdot y_i + \\
&\quad - log(1 - s_t^{'jl}) \cdot c_t^j \cdot (1 - y_i) \\
f(c_t^j) &= -log(s_t) \cdot c_t^j - log(1 - s_t) \cdot (1 - c_t^j) \\
f_y(\hat{y}_i^j) &= -y_i \cdot log(\hat{y}_i^j)
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{L}_{enc} &= \sum_{j=1}^{J} P^j \Big[\mathcal{D}^j f_y(\hat{y}^j) + (1 - \mathcal{D}^j) f_y(\tilde{y}^j)\Big] \\
\mathcal{L}_{sel} &= \sum_{j=1}^{J} -\mathcal{D}^j * sc\hat{o}res^j * log(scores^j) \\
\mathcal{L}_{gen} &= \sum_{j=1}^{J}\sum_{t=1}^{T} \Big[\mathcal{D}^j f(\hat{c}_t^j) - (1 - \mathcal{D}^j) f(\tilde{c}_t^j)\Big] + \\
&\quad \sum_{j=1}^{J}\sum_{t=1}^{T}\sum_{l=1}^{L} \Big[\mathcal{D}^j f(\hat{s}_t^{jl}) - (1 - \mathcal{D}^j) f(\tilde{s}_t^{jl})\Big]
\end{aligned} \tag{6}
$$

The final joint objective and the expected cost is defined as:

$$\mathcal{L}(\boldsymbol{r}, \boldsymbol{x}, e_1, e_2, y) = \mathcal{L}_{gen} + \mathcal{L}_{sel} + \mathcal{L}_{enc}$$

$$\min_{\theta e \theta g \theta s} \sum_{(\boldsymbol{x}, e_1, e_2, y) \in N} \mathbb{E}_{\boldsymbol{r} \sim sel(\boldsymbol{c}), \boldsymbol{c} \sim gen(\boldsymbol{x})} \mathcal{L}(\boldsymbol{r}, \boldsymbol{x}, e_1, e_2, y) \tag{7}$$

where θe, θg, θs are the parameters used in the Encoder, Generator and Selector respectively.

5 Experiments

5.1 Dataset

We evaluate our approach by performing experiments on the SemEval 2010 Task 8 dataset. A collection of sentences is provided with marked target entities and ground-truth for the relation between the entities.

The dataset contains 10,717 sentences, where 8000 entries are used in training. There are 9 types of relationships plus an “Other” class (Table 1). Relationship types (except for “Other”) are expressed bi-directionally, making 19 classes in total. Following SemEval 2010’s protocol, we used the macro-F1 metric in the experiment, excluding all cases of the “Other” class.

Besides the original SemEval 2010 Task 8 data, we asked 3 human annotators to label rationales used in the test set. Annotators were asked to choose 0-3 rationales for each test sentence based on their intuition. Each annotator is asked to roughly annotate one-third of the test set. Distribution of rationales are as follows: no-rationales:2%, 1-rationale:57%, 2-rationales:38% and 3-rationales:2%. Note that we did not annotated the training set.

Cause-Effect	Component-Whole	Content-Container
Entity-Destination	Entity-Origin	Instrument-Agency
Member-Collection	Message-Topic	Product-Producer

Table 1: Relation classes in SemEval 2010 task 8

5.2 Experimental Setup

We now describe in detail the parameters used in our experiments. We initialize the word vectors with the GoogleNews[2] corpus. The CNN filter size was initialized to 50. We use at most 3 rationales and set the penalizing factor P_j to 0.1 based on our analysis of the training set. Due to the lack of sufficient samples in certain relation classes, we ignored the relationship directionality while computing the Relation heuristic. Finally, the dimensionality of the POS tag vectors and the position vectors is set to 25, and the Generator is set to ignore nouns, articles and adjectives.

[2] https://code.google.com/p/word2vec

Finally, after training for 10 iterations, the Generator includes the output from the Corpus and Rationale heuristics. The K frequently used words in the Corpus heuristic are computed for each relation class, where the threshold for K_l is 10% of number of sentences of the given l relation class. The Rationale heuristic selects the top 25% of words, hw^l, output by the Word heuristic.

We used the Adagrad optimizer in our experiments and the learning rate is initialized to 0.01, and reduces by 10% after every iteration. During each training iteration, we sample 25 times from the Generator, and re-sample 25 times when training the Selector and the Encoder.

5.3 Results

We present a comparison of relation classification between our model and the state-of-the-art models in Table 2. Dependency tree models are neural network models that work with word dependency tree parsed by a grammar parser. Each model utilizes the dependency tree differently. For instance MVRNN (Socher et al., 2012) uses the whole dependency tree, while SPTree (Miwa and Bansal, 2016) experiments using the entire tree and the nodes on the path that connect the target entities. We refer to the models that do not use a dependency tree as independent models.

Classifier	F1
Manually Engineered Models	
SVM(Rink and Harabagiu, 2010)	82.2
Dependency Tree Models	
RNN (Socher et al., 2012)	77.6
MVRNN (Socher et al., 2012)	82.4
FCM (Yu et al., 2014)	83.0
Hybrid FCM (Gormley et al., 2015)	83.4
DepNN (Liu et al., 2015)	83.6
DRNNs (Xu et al., 2015)	85.6
SPTree (Miwa and Bansal, 2016)	84.5

Classifier	F1
Independent Models	
CNN & softmax (Zeng et al., 2014)	82.7
Stackforward(Nguyen and Grishman, 2015)	83.4
Vote-bidirect (Nguyen and Grishman, 2015)	84.1
Vote-backward (Nguyen and Grishman, 2015)	84.1
ATT-Input-CNN (Wang et al., 2016)	87.5
ATT-Pooling-CNN (Wang et al., 2016)	88.0
Rationale Models	
Proxy-Rationale-1(Hsu et al., 2018)	84.5
Proxy-Rationale-2(Hsu et al., 2018)	85.3
Proxy-Rationale-3(Hsu et al., 2018)	87.8
GAN(Hsu et al., 2018)	89.5
Our Model	**90.7**

Table 2: Comparisons with benchmarking models

Table 3, presents a detailed comparison between the different variations of our model and other rationale-based models. We evaluate the rationale quality using macro-F1, where precision and recall are computed by taking the intersection between the generated and the manually labeled rationales.

Rationale Models	Relation F1	Rationale F1
Proxy-Rationale-1(Hsu et al., 2018)	84.5	6.3
Proxy-Rationale-2(Hsu et al., 2018)	85.3	22.2
Proxy-Rationale-3(Hsu et al., 2018)	87.8	26.1
GAN(Hsu et al., 2018)	89.5	33.2
Our Model prior global heuristics s^{global}	89.5	37.1
Our Model after global heuristics s^{global}	**90.7**	**53.2**

Table 3: Comparisons between variations and with rationale models

Finally, we summarize a few samples of rationales chosen from the test set in Table 4. We observe that the extracted rationales are similar to the rationales chosen manually and they can be used to infer the relationship between entities.

Relation Class	Selected rationales with target entities
Cause-Effect	dips$_{e1}$ caused by ... y-rays$_{e2}$, liver$_{e1}$ ... cause ... hypertension$_{e2}$ plaintiffs$_{e1}$ ...resulting from ... explosion$_{e2}$, storm$_{e1}$ generated by ... cold$_{e2}$
Component-Whole	site$_{e1}$ is part of ... network$_{e2}$, cabinet$_{e1}$ encloses ... woofer$_{e2}$, ladder$_{e1}$ comprises ... steps$_{e2}$, crocodile$_{e1}$ has ... snout $_{e2}$
Content-Container	guns$_{e1}$ are locked in safe$_{e2}$, grenade$_{e1}$ hidden inside canister$_{e2}$ spider$_{e1}$ was contained in ... box$_{e2}$, suitcase$_{e1}$ full ... books$_{e2}$
Entity-Destination	weapons$_{e1}$ ... delivered to ... navy$_{e2}$, money$_{e1}$ ... into ... custody$_{e1}$ children$_{e1}$ were handed over to relatives$_{e2}$, water$_{e1}$ ... been poured into ... river$_{e2}$
Entity-Origin	squirrel$_{e1}$ popped out of ... shirt$_{e2}$, robbers$_{e1}$ ... away from scene$_{e2}$ gases$_{e1}$ emanated from ... sources$_{e2}$, starch$_{e1}$ is source of sugars$_{e2}$
Instrument-Agency	mechanic$_{e1}$ tightens ... with spanner$_{e2}$, intellect$_{e1}$ wields pen$_{e2}$ project$_{e1}$ uses art$_{e2}$ as instrument, gives ... best practices$_{e1}$ for programmers$_{e2}$
Member-Collection	bloat$_{e1}$ of hippopotamuses$_{e2}$, formation$_{e1}$ comprised of ... dragoons$_{e2}$ surgeon$_{e1}$ is part of the team$_{e2}$, sergeant$_{e1}$ in army$_{e2}$
Message-Topic	caveats$_{e1}$ outlined ... e-mail$_{e2}$, speech$_{e1}$ ... about conversation$_{e2}$ speech$_{e1}$ was summary of ... problems$_{e2}$, parameters$_{e1}$ described text$_{e2}$
Product-Producer	production materials$_{e1}$ by industries$_{e2}$, products$_{e1}$ grown by ... defendant$_{e2}$ engineers$_{e1}$ ... devised ... method$_{e2}$, investment firm$_{e1}$ co-founded by ... head$_{e2}$

Table 4: List of words/phrases that are selected as rationales in test set by our model. Additional words are denoted as ... if they are located between entities and rationales but are not selected as rationales.

6 Discussion and Future Goals

6.1 Logistic Regressions and RNN

The major difference between this work and the previous rationale-based research is that we designed the Generator without a sequential RNN model. Specifically, we developed a layer by layer process that considers different levels of text information for three reasons. First, in relation classification, the excess sequential modeling power from RNN is not required since rationales are often an extremely small span of words. Also, rationales are often strongly-related to POS tags which can be used to replace the sequence information. Second, the purpose of rationales in relation classification is to augment features and illuminate how the model derives the answer. This is different from other tasks like aspect-level sentiment classification in Lei et al. (2016) where the rationales are treated as a compression of the input. Although more empirical experiments are required, we hypothesize that this simpler multilevel heuristic approach suits better for tasks with simpler rationales. Finally, dropping RNN can help the model parallelize and also less prone to mode collapse and gradient vanishing.

6.2 Axillary Information

The improvement in rationale interpretability comes from regularizing the Generator through a wide set of knowledge sources, like the Relation, the Corpus and the Rationale heuristic, and we believe this can be further improved by including more knowledge sources. For example, the Entity heuristic can be improved by annotating entities through a large open source data such Wikipedia, or the Corpus heuristic can be more diverse with distant supervision tricks. However, the amount of axillary information to include and to what degree is a challenge in itself.

Another major difference is that the Generator is directly related to the relation-labels through the Relation and Word heuristics, which is different from Hsu et al. (2018) where the Generator is not aware of the ground-truth. This can be a potential challenge when the diversity of relation vastly increases. To be more specific, the number of relations in some knowledge graph population tasks can be more than a few thousands, which can lead to severe label imbalance issues. Also, we notice several relation classes share the same rationales, like ‘of’ is commonly seen in Member-Collection and Component-

Whole relations. This can be unfavorable for the Word heuristic since our approach assumes rationales are more heterogeneous than homogeneous between different relations. A potential solution is to cluster or introduce hierarchical structures to the relations thereby reducing diversity.

6.3 Potential Applications: Noise-Reduction in Distant Supervision

A potential application of this work, besides classifying relations and providing interpretable results, is to help reduce noise in distant supervision for relation classification. Distant supervision is a commonly used approach in relation classification where producing ground-truth data is expensive. Distant supervision exploits a simple assumption: if a sentence s contains an entity-relation pair $(e1, relation, e2)$ that exists in a trustworthy knowledge source, then s can be a training data for the $relation$. As a result, distant supervision often contains many noisy false-negative training samples and degrades the result. To reduce the noise, one can penalize sentences that do not contain commonly used rationales for the relation.

7 Conclusion

In this paper, we have proposed an improved rationale-based model for entity relation classification. In our model, besides context word information, we also moderate rationale generation with multiple heuristics computed from different text level features. With multilevel heuristics, we successfully reduce the variability in the Generator to achieve meaningful rationales. Quantitative analysis demonstrates that our model improves both classification performance and the rationale quality. Finally, we provide an annotated test set for rationales, which can be used in future related research to evaluate rationales.

Acknowledgments

We thank the COLING reviewers for their detailed and insightful feedback, and also thank the annotators for their intensive efforts for developing the annotated set. This material is based upon work supported in whole or in part with funding from LAS. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the LAS and/or any agency or entity of the United States Government.

References

Martin Arjovsky and Leon. Bottou. 2017. Towards principled methods for training generative adversarial networks. In *arXiv preprint arXiv:1701.04862.*

Martin Arjovsky, Soumith Chintala, and Leon. Bottou. 2017. Wasserstein gan. In *arXiv preprint arXiv:1701.07875, 2017.*

Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie. Li. 2017. Mode regularized generative adversarial networks. In *ICLR*.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *NIPS*.

Bang Duhyeon and Shim Hyunjung. 2018. Improved training of generative adversarial networks using representative features. In *arxiv: preprint arXiv:1801.09195.*

Claudio Giuliano, Alberto Lavelli, and Lorenza Romano. 2006. Exploiting shallow linguistic information for relation extraction from biomedical literature. In *11th Conference of the European Chapter of the Association for Computational Linguistics.*

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NIPS*, pages 2672–2680.

Matthew R Gormley, Mo Yu, and Mark Dredze. 2015. Improved relation extraction with feature-rich compositional embedding models. In *EMNLP*, pages 1774–1784.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron. Courville. 2017. Improved training of wasserstein gans. In *arXiv preprint arXiv:1704.00028.*

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2009. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *NAACL HLT*, pages 94–99.

Shiou Tian Hsu, Changsung Moon, Paul Jones, and Nagiza Samatova. 2018. An interpretable generative adversarial approach to classification of latent entity relations from unstructured sentences. In *AAAI*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*, pages 1746–1751.

Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt. Kira. 2017. On convergence and stability of gans. In *arXiv preprint arXiv:1705.07215*.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *EMNLP*, pages 107–117.

Yang Liu, Furu Wei, Sujian Li, Heng Ji, Ming Zhou, and Houfeng Wang. 2015. A dependency-based neural network for relation classification. In *ACL*, pages 285–290.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *ACL*, pages 1105–1116.

Thien Huu Nguyen and Ralph Grishman. 2015. Combining neural networks and log-linear models to improve relation extraction. In *IJCAI Workshop on Deep Learning for Artificial Intelligence (DLAI)*.

Feng Niu, Ce Zhang, Christopher Ré, and Jude W Shavlik. 2012. Deepdive: Web-scale knowledge-base construction using statistical learning and inference. *VLDS*, 12:25–28.

Bryan Rink and Sanda Harabagiu. 2010. Utd: Classifying semantic relations by combining lexical and semantic resources. In *SemEval*, pages 256–259.

Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. 2017. Stabilizing training of generative adversarial networks through regularization. In *NIPS*.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training gans. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2234–2242.

Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *EMNLP*, pages 1201–1211.

Akash Srivastava, Lazar Valkov, Chris Russell, Michael U. Gutmann, and Charles Sutton. 2017. Veegan: Reducing mode collapse in gans using implicit variational learning. In *NIPS*.

Linlin Wang, Zhu Cao, Gerard de Melo, and Zhiyuan Liu. 2016. Relation classification via multi-level attention cnns. In *ACL*, pages 1298–1307.

Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2015. Semantic relation classification via convolutional neural networks with simple negative sampling. In *EMNLP*, pages 536–540.

Mo Yu, Matthew Gormley, and Mark Dredze. 2014. Factor-based compositional embedding models. In *NIPS*, pages 95–101.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *COLING*, pages 2335–2344.

Ye Zhang, Iain Marshall, and Byron C Wallace. 2016. Rationale-augmented convolutional neural networks for text classification. In *EMNLP*, pages 795–804.