Integrating User Rules into Neural Text Inference

[1]Wei Zheng, [1]Yuqing Sun*, [2]Yexin Zhang, [1]Bin Gong
[1]School of Software, Shandong University, Jinan, China
[2]Global Energy Interconnection Research Institute Co., Ltd, Beijing, China
zw1995@mail.sdu.edu.cn, sun_yuqing@sdu.edu.cn, yexinzg@163.com, gb@sdu.edu.cn

***Abstract*—Incorporating user-defined heuristic rules into neural text inference methods has the potential to align models with user intentions and domain knowledge, thereby improving interpretability. In this study, we introduce a novel rule pattern that includes both domain-specific keywords and the logical relationships between keywords, which can be defined by users. We propose an approach to integrate explicit rule-based reasoning with the semantic modeling capabilities of neural networks. Specifically, our method employs a parallel framework wherein a neural classifier is trained on labeled text data for prediction, while a Semantic-Logic Network (SLN) forms rule inference as a satisfiability problem. We use a Jensen-Shannon (JS) loss to ensure consistent predictions on both sides for mutual regularization. The experiment results show that our approach outperforms baseline methods. We also did ablation analysis on our method, it shows that the performance of both the SLN and the classifier contribute to the final results. Additionally, for the case that lacks explicit user rules, we propose a boosting method to automatically generate rules from labeled texts which is beneficial for text inference and improve the model performance.**

***Index Terms*—User Rules, Semantic-Logic, Text Inference**

I. Introduction

Integrating user-defined rules into neural text inference model is a highly researched topic, as it offers two key benefits. First, user rules enhance the model performance by regulating the learning process. The model learns not only from labeled data but also from the prior knowledge within the rules [1]. Second, rule-based inference improves the model interpretability. It ensures a fully transparent and faithful decision process [2]. This is important for the tasks such as Text Subscription and Text Review, where users express their preferences on text subscription by rules and require the explantions for the text inference results.

Existing methods for integrating user rules treat rule-based inference as a probability satisfaction problem. Rules are transformed into real values by the techniques such as soft logic and fuzzy t-norm [1], [3], where these values represent the probability of rule satisfaction. Maximizing the probability of rule satisfaction is also as an additional objective. However, encoding rules as real values often loses their semantic meanings, retaining only syntactic information [4]. Another approach maps rules into the semantic space of text to assist in text inference [5]. However, these methods require task-specific rule learning objectives, such as mathematical rule rewriting [6]. Hence, it is challenging to apply to other tasks.

*Corresponding author

In this paper, we addresse a novel user rule pattern designed for the tasks such as Text Subscription and Text Review. The user rules are composed of domain-specific keywords and logical constraints between these keywords, which are employed to express user preferences on text subscription. For example, the upper part of Figure 1 provides a case of user rules for the Text Subscription task. The user is interested in emergency news happend in a specific geographical area, including *natural disasters*, *social security accidents*, etc. The subscribed texts should statisfy any of these topics. Then the user rules are formalized as keywords-level logical combination patterns with logical connectives such as conjunction and disjunction ($\wedge$, $\vee$).

To incorporate this form of user rules into neural text inference, we introduce a Semantic-Logic Network (SLN). It is designed to identify whether the text complies with user rules in a neural way. It is sequentially composed of three detection modules. The first module checks whether the text contains the keywords specified in the rules, while the second module checks the conjunction of keywords, and the third module checks the satisfaction of disjunction rules. The results from each detection module are combined based on the logical relationships defined in the rule, yielding the overall satisfaction detection results on the text. Each detection module utilizes a neural network, enabling implicit semantic encoding for simulating explicit symbolic Boolean matching. It is trainable using text-level labeled data and the base module (term detection) is pre-trained on generic data to enhance model robustness.

Additionally, we employ a parallel structure to combine the proposed SLN with another neural classifier to further enhance the inference performance. A Jensen-Shannon (JS) loss is applied to ensure the consistency between the predictions of the SLN and the classifier. In the absence of user-defined rules, we introduce a boosting method to generate rules using labeled data. We use information gain to identify the important keywords and subsequently explores logical combinations to form rules. The current rule is evaluated using the Boolean search accuracy of the labeled data. The experiment results show the effectiveness of automatically generated rules compared to manually defined ones. Our main contributions are as follows:

- We introduce a novel user rule pattern that combines domain-specific keywords and logical combinations.
- We propose a Semantic-Logic Network for integrating

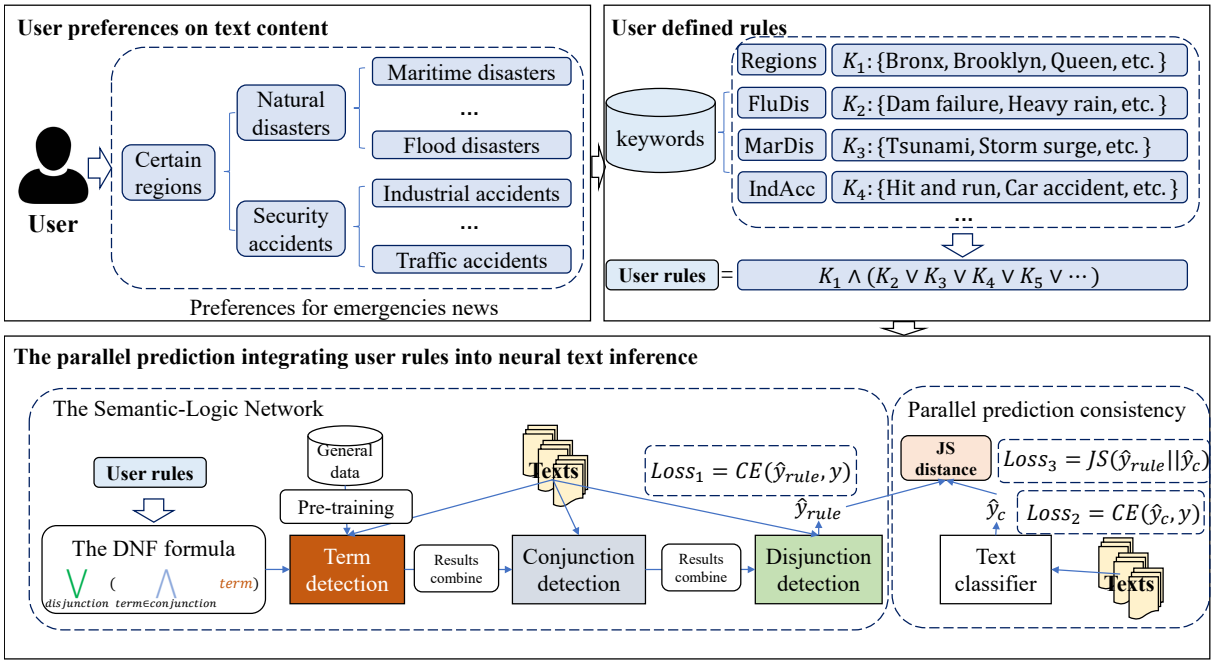


Fig. 1. User expresses news preference with keyword-level logic rules. The block below illustrates the framework of our approach, integrating the user defined rules into neural text inference.

user rules into neural text inference. The use of a parallel structure and JS loss ensures the prediction consistency.

- Experimental results indicate that our model outperforms baseline methods on multiple tasks. We also provide a detailed analysis of the impact of rules on the model performance.

II. Related Work

The most relevant works are the text classification methods, which directly classify texts of interest and disinterest to users based on a large amount of labeled data. Examples include convolutional neural networks (CNNs) [7], [8], recurrent neural networks (RNNs) with attentions or gates [9]–[11], and pretrained language model-based methods [12]–[16]. While these methods perform well provided with sufficient labeled data, they often lack explainability.

The representative methods for incorporating rules often model rule inference as the probabilistic logic satisfiability problem. Techniques such as fuzzy t-norm [1] and probabilistic soft logic [3] are employed to convert the rules into real values, representing the probabilities of rule satisfaction. Then, additional logical constraints are introduced by maximizing the probabilities of rule satisfaction during model training. However, these methods often sacrifice semantic meanings for syntax [4]. Some researchers treat logic rules as trainable representations [6], [17]. However, these methods require task-specific rule learning objectives. Hence, it is challenging to apply to our tasks, where there is a more flexible user rule pattern that contains domain-specific keywords and their logical relationships.

Our work is also related to methods that use the consistency constraints to improve prediction performance on various tasks. The consistency constraints are typically applied between two models with the same prediction objective but different views, such as between extractive and abstractive summarization models [18], between sentence-level and word-level category prediction models [19], and between classifiers based on source text and summary [20]. Prior research has demonstrated that enforcing consistency constraints during model training can effectively improve model performances [21].

III. Method

A. Problem Definition and Framework

The **user-rule** is an expression F that defines logical relationships between keywords, using logical connectives $\vee, \wedge$. Each term K represents a set of keywords that are either topic-related or semantically correlated with each other. The formula F can be converted into an equivalent Disjunctive Normal Form (DNF). In DNF, a simple conjunctive formula r consists of terms represented as $r = \bigwedge_{i=1}^{\kappa} K_i$, where κ is the number of terms in r. The DNF is obtained by taking the disjunction of simple conjunctive formulas, denoted as $DNF = \bigvee_{j=1}^{\tau} r_j$, where τ is the number of r. The set of all simple conjunctive formulas in the user rule is denoted as $R = \{r_1, r_2, ..., r_\tau\}$. For simplicity, in the following text, we refer to a simple conjunctive formula r as a rule.

The task of text inference is to determine whether a given text $x \in X$ satisfies the user rule set R. Due to the nature of DNF, satisfaction of any $r \in R$ is sufficient. The inference model provides a text-level prediction probability $\hat{y} \in (0, 1)$

and a rule-level prediction probability set $\{\hat{y}_1^r, \hat{y}_2^r, ..., \hat{y}_\tau^r\}$. Each $\hat{y}_i^r \in (0,1)$ represents the probability that the text x satisfies the rule r_i. The detected rules offer an explanation of the input text to the user. Our task framework is illustrated in Figure 1, which includes a parallel structure that combines the proposed SLN with a neural classifier using a mutual JS loss.

B. Semantic-Logic Network

The SLN is designed to determine whether a text satisfies the defined semantics by keywords and their combinations in the given rule. It is composed of three sequentially arranged detection networks.

1) Term Detection: To determine if the text x contains the meanings of a keyword $k \in K$ in the given rule, we calculate the relevance between each word in x and each keyword $k \in K$, by computing the dot product of their pretrained embedding vectors. The embedding matrices for the text is denoted as $\boldsymbol{x_e}$ and the keywords vector is denoted by $\boldsymbol{K}$ (Section IV-A2). The resulting vector $\boldsymbol{a}$ is obtained through matrix multiplication: $\boldsymbol{a} = \boldsymbol{x_e} \times \boldsymbol{K}$. Next, $\boldsymbol{a}$ is concatenated with the text representation vector $\boldsymbol{x}$ for information integration, where $\boldsymbol{x}$ is obtain by a lightweight network TextCNN [7]. We employ an MLP network to amalgamate information from x and K while reducing dimension, resulting in the vector $\boldsymbol{t} = MLP(\boldsymbol{x} \oplus \boldsymbol{a})$.

The predicted result $\hat{y}^t = \sigma(\boldsymbol{W}\boldsymbol{t} + \boldsymbol{b})$ is evaluated against the true label y^t using the cross-entropy loss function: $L_t = cross\_entropy(y^t, \hat{y}^t)$. The true label y^t is obtained through string matching between the text and keywords.

2) Conjunction Detection: To verify if the text x satisfies the conjunctive fomular $r \in R$, we leverage the information in the vector $\boldsymbol{t}$, which merges the information of the text and the keywords, to combine the term detection results. We employ a CNet, which is a three-layer MLP, to capture the fusion information of x and r. The input to CNet is the concatenated vector of $\mathbf{t_i}$ corresponding to each term t_i contained in rule r. The terms in r have conjunctive relationships with each other.

$$\boldsymbol{r} = CNet_{|K_* \ in \ r|}(\boldsymbol{t}_1 \oplus \boldsymbol{t}_2 \oplus ...) \tag{1}$$

The evaluation is performed by calculating the cross-entropy loss, denoted as $L_r = cross\_entropy(y^r, \hat{y}^r)$, between the predicted probability $\hat{y}^r$ and the true label y^r. The true label y^r is obtain through a Boolean check on x using the rule r.

3) DNF Prediction: Similarly, we employ a DNet to determine whether the text satisfies the final rule in the DNF form. The DNet takes the concatenated vectors $\boldsymbol{r}$ as input, allowing the network to capture the interactions between the rule features. The output of the DNet is then compared to the ground truth label y of the text x to evaluate the satisfaction of the DNF. The loss is $L_R = cross\_entropy(y^R, \hat{y}^R)$, where:

$$\hat{y}^R = DNet_{|r_* \in R|}(\boldsymbol{r}_1 \oplus \boldsymbol{r}_2 \oplus ...) \tag{2}$$

4) Parallel Prediction: To improve the model performance, we adopt a parallel structure that combines the SLN and a neural classifier. The text is simultaneously provided as input into both SLN and the classifier for prediction. To ensure the consistency of their predictions, we calculate the Jensen-Shannon (JS) distance between the output predictions of the classifier ($P(x)$) and the SLN ($Q(x)$). The JS distance, denoted as $JS(P||Q)$, is a variant of the Kullback-Leibler (KL) divergence that addresses the asymmetry issue. It quantifies the difference between two probability distributions. Specifically, the JS distance is calculated as the average of two KL divergences:

$$JS(P||Q) = \frac{KL(P||\overline{P,Q}) + KL(Q||\overline{P,Q})}{2} \tag{3}$$

To maintain the consistency, we incorporate the JS distance as a regularization term in the joint loss for fine-tuning. The joint loss L_u is defined as follows, where hyperparameters $\alpha \in (0,1)$ and $\beta \in (0,1)$ control the tradeoff between the neural classifier and the SLN:

$$L_u = \alpha L + \beta L_R + (1 - \alpha - \beta) JS(P||Q) \tag{4}$$

5) Training Process: The term detection module is initially trained on a general corpus [22]. Subsequently, the conjunction detection module is added and trained under the given conjunctive rules. All parameters are updated during this process, while L_t is suspended.

Subsequently, the disjunction detection module is added for further training on the text-level labeled data. The previous losses (L_t and L_r) are suspended, and the SLN, with all parameters updatable, is trained using the loss function L_R.

The classifier is pre-trained on labeled text data. Then the pre-trained SLN and the pre-trained classifier are combined for further fine-tuning. L_u is used as the loss for this process.

C. Automatic Generation of Rules

To automatically generate rules in the absence of user-defined rules, we propose a two-stage process using the labeled text dataset. In the first stage, we extract the user preferences from the dataset by identifying important keywords. Unlike traditional methods that operate at the text level, our approach uses Information Gain (IG) to identify the important keywords to distinct positive and negative samples.

Let $V = \{v_1, v_2, ..., v_{|V|}\}$ represent the vocabulary corresponding to the sample set X, where $|V|$ denotes the size of V. The information entropy $H(y)$ quantifies the uncertainty of the label y: $H(y) = -\sum_{y=0,1} p(y) \log p(y)$. For a given word $v \in V$, the variable $e_v \in \{0,1\}$ indicates whether the text x contains the word v. Here, $e_v = 1$ (denoted as e_{v1}) indicates the presence of a word v in x, while $e_v = 0$ (denoted as e_{v0}) indicates its absence. The conditional entropy $H(y|e_v)$ quantifies the uncertainty of the label y when the existence of v is known. The conditional entropy $H(y|e_{v1})$ is calculated as: $H(y|e_{v1}) = -\sum_{y=0,1} p(y|e_{v1}) \log p(y|e_{v1})$. Similarly, we have $H(y|e_{v0})$. The overall conditional entropy $H(y|e_v)$ and the information gain of a word v is as follows:

$$H(y|e_v) = -\sum_{e_v=0,1} p(e_v) H(y|e_v) \tag{5}$$

TABLE I
EXAMPLES OF THE *emergency* DATASET.

Content	Label
Typhoon *lichema* landed in (**exact region**). The meteorological observatory continues to issue a Red **Rainstorm** warning.	1
(**exact region**) was ravaged by *lichema*. After the **typhoon**, seafood was picked up casually, and a group of fish "bouncing".	0
Affected by **Typhoon** *lichima*, it has **rained** for many days in (**exact region**), and now it has ushered in a small sunny day.	0

$$IG(y, e_v) = H(y) - H(y|e_v) \tag{6}$$

In the second stage, we search for the combinations of the high-information-gain keywords to form the conjunctive rules. The current rule is evaluated by the Boolean search accuracy on the labeled data. An evaluation score s is defined to quantify the suitability of the current rule:

$$s = \frac{\sum_i \mathbb{I}(y_i^l = y_i)}{|X|} \tag{7}$$

where y_i^l is the Boolean matching results on text x_i using the current rule. We select the top-scoring conjunctive rules to construct the DNF. By eliminating unnecessary keywords, our method optimizes the search space. The complexity for keyword extraction is $O(|X| \cdot |V|)$ and for conjunctive rule generation is $O(|X|)$.

IV. EXPERIMENTS

A. Text Subscription

1) Dataset: The *emergency* dataset focuses on the emergencies news occurring in a certain region, covering topics such as $natural\ disasters$, $societal\ governance$, $production\ safety\ incidents$ and etc. The dataset comprises 80,000 samples, with an equal distribution of positive and negative samples. The text samples are padded to a maximum length of 400 tokens.

In Table I, both positive and negative samples contain keywords like $typhoon$, $rainstorm$, and the specific region. However, positive samples provide the essential semantic in the context characterizing emergencies, while negative samples not. Therefore, it is necessary to consider both keywords and context semantics in handling the above task. For the rules in the *emergency* dataset, Figure 1 provides a breif version of example that does not include some subdomains such as *Meteorological Disasters*.

2) Model Implementation: For the SLN model, the term and the rule embedding dimensions are set to 200. We use the Adam optimizer with a dropout ratio of 0.5 at each fully connected layer. The JS ratio is set to 0.2, while $\alpha = \beta = 0.4$.

The vector $\mathbf{K}$ is the average of all keywords in the set K. This captures the shared semantic features among synonyms, which are often closely related in the semantic space. Although there should be a logical $\vee$ relationship among keywords within a set, expanding the rules in this way would result in exponential complexity, which is related to the size of the set.

TABLE II
MODEL COMPARISON RESULTS ON *emergency* AND SAR.

Model	*emergency*			*SAR*		
	prec.	rec.	F1	prec.	rec.	F1
Boolean Search	77.0	74.5	75.7	81.6	76.6	79.0
TextCNN	94.0	93.6	93.8	92.7	96.7	94.7
BiLSTM	95.5	91.9	93.7	92.9	96.6	94.7
BiLSTM-2DCNN	95.2	93.7	94.4	94.3	96.6	94.7
HAN	95.7	93.6	94.6	-	-	-
LSTM-Capsual	95.7	93.2	94.4	94.3	96.7	95.4
BERT	96.8	94.5	95.6	95.6	96.7	96.1
RoBERTa	96.4	94.7	95.5	95.7	96.9	95.8
SLN	94.6	94.2	94.4	92.6	97.6	95.0
SLN+BERT	**98.4**	**98.4**	**98.4**	**97.2**	**97.9**	**97.5**

Additionally, for geographical keyword sets that include place names, to mitigate embedding issues arising from diverse geographic names, we replace all subordinate geographic names with their superior counterparts. For instance, if a user is interested in events occurring in the $New\ York$ region, then $Brooklyn$ would also satisfy the condition. Therefore, all subordinate regions under $New\ York$ are replaced with $New\ York$.

3) Comparision Methods: We compare our approach with the traditional **Boolean Search** baseline, which performs a hard matching of the text and the given rule. We also adopt some advanced text classification baselines, each utilizing different structures for text embedding. The **TextCNN** [7] approach employs convolutional kernels of various heights to extract features from the text. **BiLSTM** integrates two layers of LSTM [9] in opposite directions to capture sequential information. **BiLSTM-2DCNN** [23] utilizes two-dimensional convolution and pooling operations to derive a text representation. **HAN** [10] combines word-level and sentence-level BiLSTM to capture hierarchical information. **LSTM-CAPSUAL** [24] encodes the input text using an LSTM layer and employs capsule networks for classification. For the Pre-trained Language Models based methods, we compared with **BERT** [12] and **RoBERTa** [15] models, which are fine-tuned on the *emergency* dataset.

4) Results and Analysis: Table II shows the inference results on the *emergency* dataset. All the results are the average of 10-fold experiments. The upper block includes the results of baselines trained solely on *emergency*, while the lower block comprises the results of our parallel models that combine BERT [14] with SLN.

The results of the proposed parallel model demonstrate a notable improvement of 4% in accuracy and 4.2% in recall compared to the individually trained SLN. The incorporation of the JS distance in the joint loss improves the consistency of the predictions between the classifier and the SLN. Moreover, the SLN enables more precise handling of user concerns by separately detecting conjunctive rules. For the hyperparameters, we observed that changing the rule embedding sizes within the range of 150-250 had trival impacts on performance.

TABLE III
RESULTS (%) OF ABLATION TESTS ON THE SLN MODULES.

Model	SLN		Parallel	
	acc.	F1	acc.	F1
$+min+max_{w\backslash o}$	75.9	69.2	93.1	82.4
$+CNet+max_{w\backslash o}$	76.1	74.9	94.9	94.8
$+CNet+DNet_{w\backslash o}$	-	-	96.3	96.2
$+min+max_{w\backslash}$	94.1	94.0	95.6	95.8
$+CNet+max_{w\backslash}$	**94.4**	**94.4**	98.1	98.1
$+CNet+DNet_{w\backslash}$	**94.4**	**94.4**	**98.4**	**98.4**

However, the JS setting had a more substantial influence. The accuracy varied by up to 2% within the JS range of 0.2-0.8.

5) Ablation Study: In the ablation study, we evaluate the parts of SLN, particularly focusing on CNet and DNet. We also evaluate the effect of fine-tuning SLN based on the text-level supervision. Specifically, we compare the performance of CNet and DNet implemented with MLP versus min/max functions. The min function outputs the minimum detection probability of all terms belonging to the conjunctive formula, while the max function outputs the maximum detection probability of all conjunctive formulas within the DNF. Additionally, we compare the performance of the SLN fine-tuned under the labeled text ($w/$) with the SLN without fine-tuning (w/o), where the latter directly outputs the disjunction detection results $\hat{y}^R$ as the inference result.

Table III shows the results of the ablation study. The $+min$ model refers to replace the CNet with the min function, while the $+max$ model is to replace the DNet with the max function. The $+min+max$ model only adjusts the parameters in the term detection module during training. From the results, we can see that the $+CNet+DNet$ model achieves the highest average accuracy and F1 score with stable performances. However, the $+min+max$ model offers a more concise structure and comparable results, making it an attractive alternative considering the training cost.

6) Analysis of The Parallel Structure: We conducted an analysis of the impact of parallel training on SLN and the neural classifier (TextCNN [7] in this case). We provide statistics on the number of samples predicted by the model in different probability intervals in Figure 2. We employ False Prediction Masking (FPM), where the red box signifies the number after removing False Negatives, and the green box represents the number after removing False Positives, as shown in Figure 2(b), (c), and (d).

Comparing Figure 2(a) with (b), there is a notable improvement in recall after joint training. From Figure 2(a) to (c), the SLN exhibits higher confidence in assigning low probabilities (0-0.25) due to its logical inference certainty. From Figure 2(a) to (d), TextCNN demonstrates high confidence in assigning high probabilities (0.75-1) because of its ability to capture semantic information. After joint training, TextCNN assigns higher probabilities to positive samples (0.5-0.75) to (0.75-1), indicating increased confidence in True Positive predictions. The SLN assigns lower probabilities to negative samples (0.25-0.5) to (0-0.25), demonstrating higher confidence in True Negative predictions.

TABLE IV
THE EXPERT RULES OF THE SAR TASK.

$K_\#$	Content	$K_\#$	Content
K_1	{has no right etc.}	K_6	{handle, etc.}
K_2	{shareholder, etc.}	K_7	{without consent, etc.}
K_3	{actual investor.}	K_8	{loss, deficit., etc.}
K_4	{equity, etc.}	K_9	{compensate, etc.}
K_5	{pledge, etc.}		
r_1^ϵ	$K_1 \wedge ((K_2 \wedge K_3) \vee K_7) \wedge K_5 \wedge K_8 \wedge K_9$		
r_2^ϵ	$K_1 \wedge ((K_2 \wedge K_3) \vee K_7) \wedge K_4 \wedge K_5 \wedge K_8 \wedge K_9$		
r_3^ϵ	$K_1 \wedge (K_2 \vee K_3 \vee K_7) \wedge (K_5 \vee K_6)$		

TABLE V
INFERENCE RESULTS (%) WITH DIFFERENT RULES.

Rules	Parallel Predictions			
	prec.	rec.	F1	acc.
r_1^ϵ	95.0	97.3	96.1	96.1
r_2^ϵ	95.6	96.7	94.9	96.1
r_3^ϵ	**95.7**	**98.2**	**96.9**	**96.9**
r_1^ι	90.6	95.1	92.8	92.6
r_2^ι	92.5	92.9	92.6	92.6
r^μ	94.5	96.2	95.3	95.3
r^ν	95.6	97.9	96.7	96.7

B. Text Review Task

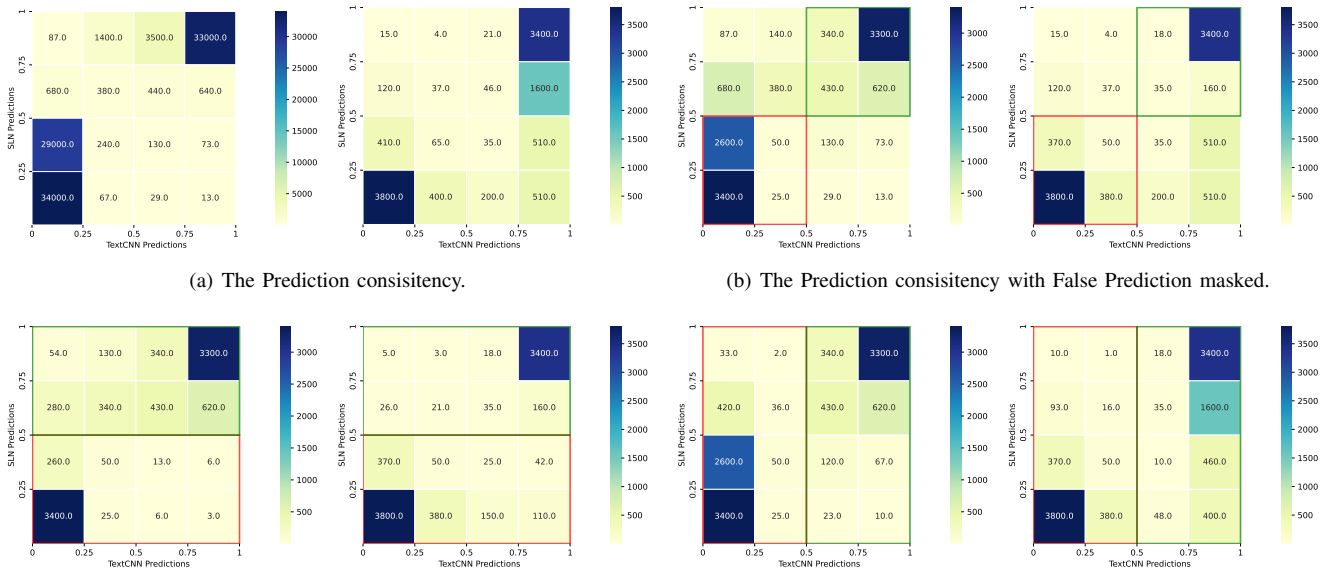
We evaluate our proposed method on another task called Subjective Answer Grading (SAG). This task is to determine whether a student answer text, in response to a subjective question, meets a specific scoring criteria. The scoring criteria are formalized as rules defined by the graders, containing keywords and their logical relationships, which describe the knowledge points required for subjective questions.

1) Dataset and Rules: For this task, an English version of the dataset subset is publicly available[1]. In Table IV, we present three rules provided by the different graders.

2) Results and Analysis: The results are presented in Table II, where r_3^ϵ is used as the user-defined rules. Our method outperforms BERT in this task, demonstrating the effectiveness of our approach in evaluating both logical matching and semantic relevance between student answers and the scoring criteria.

To evaluate the effectiveness of our rule generation method, we applied rules generated by various methods to both the SLN and parallel inference and compared their performance. The rules for comparison include expert-provided rules ($r^\epsilon*$), rules generated by randomly extracting and combining keywords ($r_{1,2}^\iota$), rules generated using Boolean search indicators (r^μ), and rules generated by our method (r^ν). Table V shows the comparison results. From the results, it is evident that rules generated by our method closely approach the performance of expert-provided rules in terms of model performance.

[1]http://splab.sdu.edu.cn/xscg/sjjydm.htm



(a) The Prediction consisitency.

(b) The Prediction consisitency with False Prediction masked.

(c) The Prediction consisitency with False Prediction masked on SLN.

(d) The Prediction consisitency with False Prediction masked on TextCNN.

Fig. 2. Statistical analysis of the prediction consistency. In each subfigure, the left image represents the independently trained models, while the right image represents the jointly trained models.

V. Conclusion

Our approach integrates user-defined rules with neural text inference to enhance the model performance and interpretability. We employ a parallel framework that includes a neural classifier and the proposed Semantic-Logic Network to address the problem of combining explicit rule-based reasoning with implicit semantic inference. We introduce a JS loss on the parallel training to ensure the consistent predictions, which is the mutual regularization of the models. In the absence of explicit rules, we employ a boosting strategy to generate valuable rules from labeled texts. Experimental results demonstrate that our approach outperforms the baseline methods on tasks such as Text Review and Text Subscription. And the generated rules helpful on improving the model performance.

VI. Acknowledgment

This work was supported by the National Natural Science Foundation of China (62376138) and the Innovative Development Joint Fund Key Projects of Shandong NSF (ZR2022LZH007).

References

[1] Z. Hu, X. Ma, Z. Liu, and et al., "Harnessing deep neural networks with logic rules," in *ACL, 2016*.

[2] S. Chaudhury, S. Swaminathan, D. Kimura, and et al., "Learning symbolic rules over abstract meaning representations for textual reinforcement learning," in *ACL, 2023*.

[3] M. Wu, W. Wang, and S. J. Pan, "Deep weighted maxsat for aspect-based opinion extraction," in *EMNLP, 2020*, pp. 5618–5628.

[4] J. Xu, Z. Zhang, T. Friedman, and et al., "A semantic loss function for deep learning with symbolic knowledge," in *ICML, 2018*, vol. 80, pp. 5498–5507.

[5] R. Zhang, Y. Yu, P. Shetty, L. Song, and C. Zhang, "Prboost: Prompt-based rule discovery and boosting for interactive weakly-supervised learning," *arXiv preprint arXiv:2203.09735*, 2022.

[6] D. Lee, C. Szegedy, M. N. Rabe, and et al., "Mathematical reasoning in latent space," in *ICLR, 2020,*.

[7] Y. Kim, "Convolutional neural networks for sentence classification," in *EMNLP, 2014*, pp. 1746–1751.

[8] R. Johnson and T. Zhang, "Deep pyramid convolutional neural networks for text categorization," in *ACL, 2017*, pp. 562–570.

[9] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[10] Z. Yang, D. Yang, C. Dyer, and et al., "Hierarchical attention networks for document classification," in *NAACL,2016*, pp. 1480–1489.

[11] A. Vaswani, N. Shazeer, N. Parmar, and et al., "Attention is all you need," in *NeurIPS, 2017*, pp. 5998–6008.

[12] M. Tan, C. N. dos Santos, B. Xiang, and B. Zhou, "Improved representation learning for question answer matching," in *ACL, 2016*.

[13] M. E. Peters, M. Neumann, M. Iyyer, and et al., "Deep contextualized word representations," in *NAACL-HLT*, pp. 2227–2237.

[14] J. Devlin, M. Chang, K. Lee, and et al., "BERT: pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT, 2019*, pp. 4171–4186.

[15] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[16] L. Li, K. Ma, and B. Cui, "Multi-information filter encoding network for multi-label text classification," in *CSCWD 2023*.

[17] W. Y. Wang and W. W. Cohen, "Learning first-order logic embeddings via matrix factorization," in *IJCAI, 2016*, pp. 2132–2138.

[18] W. T. Hsu, C. Lin, M. Lee, and et al., "A unified model for extractive and abstractive summarization using inconsistency loss," in *ACL, 2018*, pp. 132–141.

[19] R. Wang, D. Tang, N. Duan, and et al., "Leveraging declarative knowledge in text and first-order logic for fine-grained propaganda detection," in *EMNLP, 2020*, pp. 3895–3903.

[20] H. P. Chan, W. Chen, and I. King, "A unified dual-view model for review summarization and sentiment classification with inconsistency loss," in *SIGIR, 2020*, pp. 1191–1200.

[21] X. Du, B. D. Mishra, N. Tandon, and et al., "Be consistent! improving procedural text comprehension using label consistency," in *NAACL-HLT, 2019*, pp. 2347–2356.

[22] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[23] P. Zhou, Z. Qi, S. Zheng, and et al., "Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling," in *COLING, 2016*, pp. 3485–3495.

[24] Y. Wang, A. Sun, J. Han, and et al., "Sentiment analysis by capsules," in *WWW, 2018*, pp. 1165–1174.