Self-explaining deep models with logic rule reasoning

Seungeon Lee*
KAIST School of Computing
IBS Data Science Group
archon159@kaist.ac.kr

Xiting Wang†
Social Computing Group
Microsoft Research Asia
xitwan@microsoft.com

Sungwon Han*
KAIST School of Computing
IBS Data Science Group
lion4151@kaist.ac.kr

Xiaoyuan Yi
Social Computing Group
Microsoft Research Asia
xiaoyuanyi@microsoft.com

Xing Xie
Social Computing Group
Microsoft Research Asia
xing.xie@microsoft.com

Meeyoung Cha†
IBS Data Science Group
KAIST School of Computing
mcha@ibs.re.kr

Abstract

We present SELOR, a framework for integrating self-explaining capabilities into a given deep model to achieve both high prediction performance and human precision. By "human precision", we refer to the degree to which humans agree with the reasons models provide for their predictions. Human precision affects user trust and allows users to collaborate closely with the model. We demonstrate that logic rule explanations naturally satisfy them with the expressive power required for good predictive performance. We then illustrate how to enable a deep model to predict and explain with logic rules. Our method does not require predefined logic rule sets or human annotations and can be learned efficiently and easily with widely-used deep learning modules in a differentiable way. Extensive experiments show that our method gives explanations closer to human decision logic than other methods while maintaining the performance of the deep learning model.

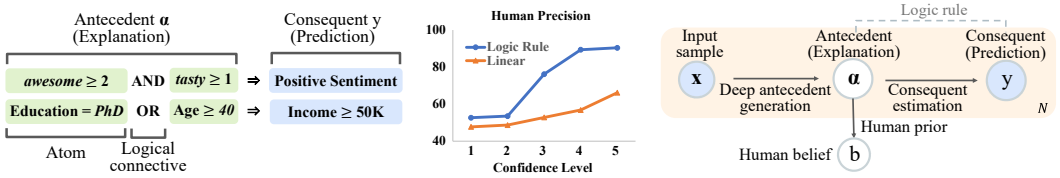
1 Introduction

Deep learning has shown high predictive accuracy in a wide range of tasks, but its inner working mechanisms are obscured by complex model designs. This raises important questions about whether a deep model is ethical, trustworthy, or capable of performing as intended under various conditions [1].

Many approaches have been proposed to help humans assess and comprehend model decisions. Recent work on explainability has primarily focused on providing **post-hoc** explanations for black-box models that have already been trained [2, 3, 4, 5, 6, 7, 8, 9, 10]. Post-hoc methods do not change the model and hence preserve the predictive performance while providing the additional benefit of explainability. These methods have achieved considerable success in providing valuable insights for model understanding, but there are also known challenges such as computational cost [11] and trust issues [12]. For example, many popular post-hoc methods test the complex black-box model thousands of times to obtain a complete and faithful understanding of the model around a single instance [1, 13, 14]. Subroutines such as full optimization or reverse propagation are generally required, introducing approximations or heuristic assumptions that may lead to misinterpretation [14, 15]. Because there is no guarantee that explanations are always faithful to the model [12], there exists a "general uneasiness" among practitioners about using and trusting post-hoc explanations [16], **Self-explaining** models naturally solve these issues, making them an ideal choice when interpretability

*Work done during internship at Microsoft Research Asia
†Corresponding Author

36th Conference on Neural Information Processing Systems (NeurIPS 2022).



(a) Rule example (b) Human precision (c) Our generative process

Figure 1: Reasoning with logic rules: (a) examples of logic rule explanations; (b) human precision for logic rule and linear regression explanations [11]; (c) generative process of our logic rule reasoning.

can be considered from the model design phase [17, 18, 19, 20]. Because the explanation mechanism is integrated inherently, these models can predict and explain simultaneously with a single forward propagation without any approximations or heuristic assumptions that decrease the faithfulness of explanations. Self-explaining methods may also improve robustness [11] and provide actionable insights for directly refining model parameters without having to calibrate the dataset [21, 22].

Based on these observations, we regard self-explaining models as providing a stronger link between humans and machine learning models, reducing misunderstanding and allowing direct control of the model based on human insights. The main challenge in achieving this new level of human-machine collaboration then becomes how to ensure self-explaining models' precision both in terms of **predictive performance** and **human precision**. Human precision refers to whether models' explanations of decision-making processes align with human decision logic. Existing approaches ensure explanations to be *easy to read*, for example, by requiring explanations to be simple and smooth in a local area [11]. However, there is little guarantee that a given explanation is a *correct* rationale for prediction according to human perception. For example, the explanation "*awesome*$\geq$2" (i.e., the word "*awesome*" appears twice in reviews) is a good rationale for positive sentiment, while "*is*$\geq$1" $\Rightarrow$ *positive sentiment* is easy to read but unreasonable to humans. Without insurance for human precision, users may constantly find unreasonable explanations, which can significantly hamper user trust and prevent them from identifying actionable insights for model refinement. An interesting research question, then, is: how can self-explaining models generate explanations that are consistent with human decision logic?

To answer this question, we need to decide what information models obtain from humans. Collecting ground-truth labels of human decision processes for every input instance [23, 24, 25, 26, 27, 28] is expensive and limits the method's scalability. Moreover, forcing the model to make decisions exactly like humans may be unwise since it could limit its data learning capability or even learn human biases that may significantly decrease the model's performance. To address this issue, it is important that humans provide guidance at a higher level that allows the models to learn freely based on data. Accordingly, we propose two desirable properties for human precision. The first property, **global coherency**, restricts the explnation form to be consistent with human reasoning logic, thereby minimizing the probability of misinterpretation. The second property, **local coherency**, requires that each explanation naturally lead to the prediction according to human perception, thereby making explanations a correct rationale for the model output. As humans can hardly provide guidance for each explanation, it is more desirable that the models can automatically guarantee local coherency based on human guidance on global coherency.

A key to satisfying these two properties is logic rules. As shown in Fig. 1a, logic rules can have flexible forms that meet human logic and preferences, making them easy to satisfy global coherency. For example, the logical connectives can be traditional (e.g., AND, OR, and NOT) or self-defined (e.g., BEFORE). Moreover, the logic rules explicitly model whether an explanation can lead to a prediction by testing the hypothesis across the entire dataset. This ensures a meaningful relationship between explanations and predictions that naturally leads to local coherency. Fig. 1b shows that logic rule explanations achieve even higher human precision than linear-regression-based explanations with local stability, while providing a confidence score that correlates with human precision (more details in Appendix A). Lastly, logic rules of different logical connectives correspond to a diverse set of feature interactions, providing the expressive power for good predictive performance.

This paper proposes **SELOR**, a framework for upgrading a deep model with a **S**elf-**E**xplainable version with **LO**gic rule **R**easoning capability. Our work is inspired by neuro-symbolic reasoning [29], which integrates deep learning with logic rule reasoning to inherit advantages from both. The most related works in this discipline are neural-guided search that finds a global logic program that works for (most) input-output pairs [19, 30, 31, 32], or identifies a local logic program and rule for the

given instance [33, 34, 35, 36, 37]. We adopt the latter paradigm, as global explanations for deep models are usually not succinct or fail to possess the same predictive power that is comparable with the deep models [14]. Existing works for generating local programs or rules have achieved promising results by effectively leveraging instance-level guidance about local programs or rules [33, 34, 35], strong external knowledge such as knowledge graphs [36, 37, 38], and a small set of predefined rules [39]. However, in our scenario, there is no instance-level guidance about the ground-truth rules, and leveraging strong knowledge such as a small set of predefined rules may introduce bias into the deep networks, as shown in our experiment results of RCN [39]. To address this, we propose a logic rule reasoning framework that leverages global level human priors about rules (e.g., desirable form and property of candidate atoms) and generate explanations by optimizing rule confidence, which can be automatically computed based on the training data. Moreover, we design a neural consequent estimator that can accurately approximate the confidence even for rare rules and combine it with recursive Gumbel-Softmax [40] to search the solution space effectively. Codes are released at Github.[3]

Our main contributions are as follows.

- Our work suggests that human precision is key for self-explaining models to bridge human logic and model decision logic seamlessly. Logic rule-based explanations enable high human precision while allowing the expressive power to achieve high prediction performance.
- We propose a logic rule reasoning framework that upgrades a given deep model into a self-explainable version by naturally integrating human priors, rule confidence modeling, and rule generation as an essential part for model prediction. Our method can achieve high human precision without depending on strong external knowledge, such as instance-level guidance about rules, knowledge graphs, or a small number of rule candidates.
- Numerical experiments and user studies confirm key strengths of our framework in terms of human precision and robustness against noisy labels with maintenance of prediction performance.

2 Deep Logic Rule Reasoning

2.1 Formulation of Logic Rules

A logic rule $\boldsymbol{\alpha} \Rightarrow y$, as shown in Fig. 1a, consists of an antecedent $\boldsymbol{\alpha}$ and a consequent y. Meanings of symbols used in this paper are defined in Appendix B.1.

- An **antecedent** $\boldsymbol{\alpha}$ is the condition to apply the rule and corresponds to an explanation in a logic form. It is represented as a sequence $\boldsymbol{\alpha} = (o_1..., o_L)$, where o_i is either an atom or a logical connective.
  - An **atom** is the smallest unit of explanation that corresponds to a single *interpretable feature* of a given input (e.g., "*awesome*$\geq$2"). The interpretable features may be different from those in deep learning models. They could, for example, have a different granularity (e.g., words or phrases) than the model features (e.g., partial words), be a statistical feature (e.g., word frequency), or be derived using external tools (e.g., grammatical tagging of a word). Mathematically, each atom o_i is a Boolean-value function, with $o_i(\mathbf{x})$ returning true if the i-th interpretable feature is present in input $\mathbf{x}$ and false, otherwise.
  - A **logical connective** combines atoms to form an explanation. Logical connectives can be traditional ones like AND, OR, NOT, or self-defined ones, as long as they take one or more Boolean values as the input and output a single Boolean value.

  We say that an input sample $\mathbf{x}$ **satisfies** an antecedent $\boldsymbol{\alpha}$, if $\boldsymbol{\alpha}(\mathbf{x})$ is true.
- The **consequent** y is the model's prediction output given the antecedent. For example, y is the predicted class in a classification task, whereas y is an explicit number in a regression task. We mainly consider classification in the paper and extend the cases to regression in Appendix B.2.

2.2 Framework for Deep Logic Rule Reasoning

Let us denote f as a deep learning model that estimates probability $p(y|\mathbf{x})$, where $\mathbf{x}$ is the input data sample and y is a candidate class. We upgrade model f to a self-explaining version by adding a latent variable $\boldsymbol{\alpha}$, which is an explanation in the logic form. Then, we can reformulate $p(y|\mathbf{x})$ as

$$p(y|\mathbf{x}, b) = \sum_{\boldsymbol{\alpha}} p(y|\boldsymbol{\alpha}, \mathbf{x}, b) p(\boldsymbol{\alpha}|\mathbf{x}, b) = \sum_{\boldsymbol{\alpha}} p(y|\boldsymbol{\alpha}) p(\boldsymbol{\alpha}|\mathbf{x}, b), \quad s.t., \quad \Omega(\boldsymbol{\alpha}) \leq S \tag{1}$$

[3] https://github.com/archon159/SELOR

Here, b represents a human's prior belief about the rules, e.g., the desirable form of atoms and logical connectives, $\Omega(\boldsymbol{\alpha})$ is the required number of logic rules to explain given input $\mathbf{x}$, and S is the number of samples (logic rules chosen by the model). Eq. (1) includes two constraints essential for ensuring explainability. The first constraint $p(y|\boldsymbol{\alpha}, \mathbf{x}, b) = p(y|\boldsymbol{\alpha})$ requires that explanation $\boldsymbol{\alpha}$ contains all information in the input $\mathbf{x}$ and b that is useful to predict y. Without the constraint, the model may "cheat" by predicting y directly from the input instead of using the explanation (more details in Appendix B.3). The second constraint $\Omega(\boldsymbol{\alpha}) \leq S$ requires that the model can be well explained by using only S explanations, where S is small enough to ensure readability ($S = 1$ in our implementation).

We can further decompose Eq. (1) based on the independence between the input $\mathbf{x}$ and the human prior belief b, following the generative process in Fig. 1c (proof and assumptions in Appendix B.3):

$$p(y|\mathbf{x}, b) = \sum_{\boldsymbol{\alpha}} p(y|\boldsymbol{\alpha}) p(\boldsymbol{\alpha}|\mathbf{x}, b) \propto \sum_{\boldsymbol{\alpha}} \underbrace{p(b|\boldsymbol{\alpha})}_{\substack{\text{Human}\\ \text{prior}}} \cdot \underbrace{p(y|\boldsymbol{\alpha})}_{\substack{\text{Consequent}\\ \text{estimation}}} \cdot \underbrace{p(\boldsymbol{\alpha}|\mathbf{x})}_{\substack{\text{Deep antecedent}\\ \text{generation}}}, \quad s.t., \quad \Omega(\boldsymbol{\alpha}) \leq S \tag{2}$$

The three derived terms correspond to three main modules of the proposed framework, SELOR:

- **Human prior** $p(b|\boldsymbol{\alpha})$ specifies human guidance regarding desirable forms for rules to minimize the probability of misunderstanding and ensure global coherency (Sec. 2.3).
- **Consequent estimation** $p(y|\boldsymbol{\alpha})$ ensures a meaningful and consistent relationship between the explanation $\boldsymbol{\alpha}$ and prediction y, so that each explanation naturally leads to the prediction according to human perception and satisfies local coherency (Sec. 2.4).
- **Deep antecedent generation** $p(\boldsymbol{\alpha}|\mathbf{x})$ uses the deep representation of input $\mathbf{x}$ learned by the given deep model f to find an explanation $\boldsymbol{\alpha}$ that maximizes global and local coherency (Sec. 2.5).

The sparsity constraint $\Omega(\boldsymbol{\alpha}) \leq S$ for the explanations can be enforced by sampling from $p(\boldsymbol{\alpha}|\mathbf{x})$. In particular, we rewrite Eq. (2) as an expectation and estimate it through sampling:

$$p(y|\mathbf{x}, b) \propto \sum_{\boldsymbol{\alpha}} p(b|\boldsymbol{\alpha})\, p(y|\boldsymbol{\alpha})\, p(\boldsymbol{\alpha}|\mathbf{x}) = \mathop{\mathbb{E}}_{\substack{\boldsymbol{\alpha}\sim\\ p(\boldsymbol{\alpha}|x)}} p(b|\boldsymbol{\alpha}) p(y|\boldsymbol{\alpha}) \approx \frac{1}{S} \sum_{\substack{s\in[1,S]\\ \boldsymbol{\alpha}^{(s)}\sim p(\boldsymbol{\alpha}|\mathbf{x})}} p(b|\boldsymbol{\alpha}^{(s)})\, p(y|\boldsymbol{\alpha}^{(s)}) \tag{3}$$

where $\boldsymbol{\alpha}^{(s)}$ is the s-th sample of $\boldsymbol{\alpha}$. For example, to maximize the approximation term with $S = 1$, the explanation generator $p(\boldsymbol{\alpha}|x)$ must find a single sample $\boldsymbol{\alpha}^{(s)}$ that yields the largest $p(b|\boldsymbol{\alpha}^{(s)})p(y|\boldsymbol{\alpha}^{(s)})$, and it needs to assign a high probability to the best $\boldsymbol{\alpha}^{(s)}$. Otherwise, other samples with a lower $p(b|\boldsymbol{\alpha}^{(s)})p(y|\boldsymbol{\alpha}^{(s)})$ may be generated, thereby decreasing $p(y|x, b)$. This ensures the sparsity of $p(\boldsymbol{\alpha}|x)$ and the model interpretability. If there are multiple best explanations that result in the exact same $p(b|\boldsymbol{\alpha}^{(s)})p(y|\boldsymbol{\alpha}^{(s)})$, the explanation generator may find all of them.

2.3 Human Prior $p(b|\boldsymbol{\alpha})$

Human prior $p(b|\boldsymbol{\alpha}) = p_h(b|\boldsymbol{\alpha})p_s(b|\boldsymbol{\alpha})$ consists of hard priors $p_h(b|\boldsymbol{\alpha})$ and soft ones $p_s(b|\boldsymbol{\alpha})$.

Hard priors categorize the feasible solution space for the rules: $p_h(b|\boldsymbol{\alpha}) = 0$ if $\boldsymbol{\alpha}$ is not a feasible solution. Humans can easily define hard priors by choosing the atom types, such as whether the interpretable features are words, phrases, or statistics like word frequency. The logical connectives to be considered (e.g., AND, NOT) can also be chosen, as well as the antecedent's maximum length L. SELOR does not require a predefined rule set. Nonetheless, we allow users to enter one if it is more desirable in some application scenarios. A large solution space increases the time cost for deep logic rule reasoning (Sec. 2.6) but also decreases the probability of introducing undesirable bias.

Soft priors model different levels of human preference for logic rules. For example, people may prefer shorter rules or high-coverage rules that satisfy many input samples. The energy function can parameterize such soft priors: $p_s(b|\boldsymbol{\alpha}) \propto \exp(-\mathcal{L}_b(\boldsymbol{\alpha}))$, where $\mathcal{L}_b$ is the loss function for punishing undesirable logic rules. We do not include any soft priors in our current implementation.

2.4 Consequent Estimation $p(y|\boldsymbol{\alpha})$

Consequent estimation ensures a meaningful and consistent relationship between an explanation $\boldsymbol{\alpha}$ and prediction y, so each explanation naturally leads to the prediction according to human perception. This is achieved by testing the logic rule $\boldsymbol{\alpha} \Rightarrow y$ across the entire training dataset to ensure that it represents a global pattern that is typically consistent with human understanding.

Empirical estimation. A straightforward way to compute $p(y|\boldsymbol{\alpha})$ is to first obtain all samples that satisfy antecedent $\boldsymbol{\alpha}$, and then calculate the percentage of them that have label y [8]. For example, given explanation $\boldsymbol{\alpha}$ ="*awesome*≥2", we obtain all instances in which *awesome* appears more than twice, and find that 90% of them have label $y =$ *positive sentiment*, then $p(y|\boldsymbol{\alpha}) = 0.9$. Large $p(y|\boldsymbol{\alpha})$ corresponds to global patterns that naturally align with human perception. Mathematically, this is equivalent to approximating $p(y|\boldsymbol{\alpha})$ with the empirical probability $\hat{p}(y|\boldsymbol{\alpha})$:

$$\hat{p}(y|\boldsymbol{\alpha}) = n_{\boldsymbol{\alpha},y}/n_{\boldsymbol{\alpha}} \tag{4}$$

where $n_{\boldsymbol{\alpha},y}$ is the number of training samples that satisfy the antecedent $\boldsymbol{\alpha}$ and has the consequent y, and $n_{\boldsymbol{\alpha}}$ is the number of training samples that satisfy the antecedent $\boldsymbol{\alpha}$.

Directly setting $p(y|\boldsymbol{\alpha})$ to $\hat{p}(y|\boldsymbol{\alpha})$ can cause two problems. First, when $n_{\boldsymbol{\alpha}}$ is not large enough, the empirical probability $\hat{p}(y|\boldsymbol{\alpha})$ may be inaccurate, and the modeling of such uncertainty is inherently missing in this formulation. Second, computing $\hat{p}(y|\boldsymbol{\alpha})$ for every antecedent $\boldsymbol{\alpha}$ is intractable, since the number of feasible antecedents A increases exponentially with antecedent length L.

Neural estimation of categorical distribution. To address the aforementioned problems, we jointly model $\hat{p}(y|\boldsymbol{\alpha})$ and the uncertainty caused by low-coverage antecedents with the categorical distribution and use a neural network to generalize to similar rules and better handle noise.

Assume that given the antecedent, y follows a categorical distribution, with each category corresponding to a class. Then, according to the posterior predictive distribution, y takes one of K potential classes, and we may compute the probability of a new observation y given existing observations:

$$p(y|\boldsymbol{\alpha}) = p(y|\mathcal{Y}_{\boldsymbol{\alpha}}, \beta) \approx \frac{\hat{p}(y|\boldsymbol{\alpha})n_{\boldsymbol{\alpha}} + \beta}{n_{\boldsymbol{\alpha}} + K\beta} \tag{5}$$

Here, $\mathcal{Y}_{\boldsymbol{\alpha}}$ denotes the $n_{\boldsymbol{\alpha}}$ observations of class label y obtained by checking the training data, and β is the concentration hyperparameter of the categorical distribution that we automatically learn with backpropagation. Eq. (5) becomes Eq. (4) when $n_{\boldsymbol{\alpha}}$ increases to ∞, and becomes a uniform distribution when $n_{\boldsymbol{\alpha}}$ goes to 0. Thus, a low-coverage antecedent with a small $n_{\boldsymbol{\alpha}}$ is automatically considered uncertain (i.e., close to uniform distribution). By optimizing Eq. (5), our method automatically balance the empirical probability $\hat{p}(y|\boldsymbol{\alpha})$ and the number of observations $n_{\boldsymbol{\alpha}}$. Probability $p(y|\alpha)$ also serves as a **confidence** score for the logic rule $\alpha \Rightarrow y$.

We then employ a neural model to predict $\hat{p}(y|\boldsymbol{\alpha})$ and $n_{\boldsymbol{\alpha}}$ to better manage noise, generalize to similar rules, and improve efficiency. In particular, we obtain A' samples of $\boldsymbol{\alpha}$ and compute $\hat{p}(y|\boldsymbol{\alpha})$ and $n_{\boldsymbol{\alpha}}$ by checking the training data. Here A' is significantly smaller than the total number of feasible antecedents A (Sec. 2.6). We use the multi-task learning framework in [41] to train the neural network with these samples. In particular, we minimize the loss in following equation.

$$\mathcal{L}_c = \frac{1}{2\sigma_p^2}||\hat{p}(y|\boldsymbol{\alpha}) - \tilde{p}(y|\boldsymbol{\alpha})||^2 + \frac{1}{2\sigma_n^2}||n_{\boldsymbol{\alpha}} - \tilde{n}_{\boldsymbol{\alpha}}||^2 + \log \sigma_p \sigma_n \tag{6}$$

$\tilde{p}(y|\boldsymbol{\alpha})$, $\tilde{n}_{\boldsymbol{\alpha}}$ are the predicted empirical probability and coverage given by the neural model, and σ_p and σ_n are standard deviation of ground truth probability and coverage. More details for training the neural network are described in the Appendix B.4 and Appendix. B.5, and effectiveness of neural consequent estimator is shown in Appendix C.5.2

2.5 Deep Antecedent Generation $p(\boldsymbol{\alpha}|\mathbf{x})$

Deep antecedent generation finds explanation $\boldsymbol{\alpha}$ by reshaping the given deep model f. Specifically, we replace the prediction layer in f with an explanation generator, so that the latent representation $\mathbf{z}$ of input $\mathbf{x}$ is mapped to an explanation, instead of directly mapping to a prediction (e.g., class label).

Given $\mathbf{z}$, which is the representation of input $\mathbf{x}$ in the last hidden layer of f, we generate explanation $\boldsymbol{\alpha} = (o_1..., o_L)$ with a recursive formulation to ensure that the complexity is linear with L (Sec. 2.6). Formally, given $\mathbf{z}$ and $o_1, ...o_{i-1}$, we obtain o_i by

$$\mathbf{h}_i = Encoder([\mathbf{z}; \mathbf{o}_1...; \mathbf{o}_{i-1}]), \quad p(o_i|\mathbf{x}, o_1..., o_{i-1}) = \frac{\mathbb{I}(o_i \in \mathcal{C}_i)\exp(\mathbf{h}_i^T \mathbf{o}_i)}{\sum_{\alpha_i'} \mathbb{I}(\alpha_i' \in \mathcal{C}_i)\exp(\mathbf{h}_i^T \boldsymbol{\alpha}_i')} \tag{7}$$

where $Encoder(\cdot)$ can be any neural sequence encoder such as GRU [42] or Transformer [43]. $\mathbb{I}$ here is the indicator function and $\mathcal{C}_i$ is the set of candidates for o_i. Every candidate should satisfy both

global and local constraints. The hard priors in Sec. 2.3 provide the global constraint and ensure that $\boldsymbol{\alpha}$ has a human-defined logic form. The local constraint requires that $\mathbf{x}$ satisfies antecedent $\boldsymbol{\alpha}$. An atom "*awesome*$\geq$ 2", for example, will be sampled only if $\mathbf{x}$ mentions "*awesome*" more than once.

We then sample o_i from $p(o_i|\mathbf{x}, o_1..., o_{i-1})$ in a differentiable way to ensure easy end-to-end training:

$$o_i = Gumbel(p(o'_i \in \mathcal{O}|\mathbf{x}, o_1..., o_{i-1})), \quad p(\boldsymbol{\alpha}|\mathbf{x}) = \prod_{i\in[1,L]} p(o_i|\mathbf{x}, o_1..., o_{i-1}) \tag{8}$$

$Gumbel$ is Straight-Through Gumbel-Softmax [40], a differentiable function for sampling discrete values. o_i is represented as a one-hot vector with a dimension of $|\mathcal{O}|$, and is multiplied with the embedding matrix of atoms and logical connectives to derive embedding $\mathbf{o}_i$.

2.6 Optimization and Complexity Analysis

Optimization. A deep logic rule reasoning model is learned in two steps. The first step optimizes the neural consequent estimator by minimizing loss $\mathcal{L}_c$ in Eq. (6). The neural consequent estimator only needs to be trained once for each dataset, and then it can be used for various deep models and hyperparameters. The second step converts deep model f to an explainable version by maximizing $p(y|\mathbf{x}, b)$ in Eq. (3) with a cross-entropy loss. This is equivalent to minimizing loss $\mathcal{L}_d = -\mathcal{L}_b(\boldsymbol{\alpha}^{(s)}) - \log\ p(y^*|\boldsymbol{\alpha}^{(s)})$, where $-\mathcal{L}_b(\boldsymbol{\alpha}^{(s)})$ punishes explanations that do not fit human's prior preference for rules (global coherency), and $\log\ p(y^*|\boldsymbol{\alpha}^{(s)})$ finds explanation $\boldsymbol{\alpha}^{(s)}$ that leads to the ground-truth class y^* with a large confidence (prediction accuracy), in which the confidence is measured by testing rule $\boldsymbol{\alpha}^{(s)} \Rightarrow y^*$ in all training data (local coherency).

Complexity analysis. Time complexity is compared in Table 1. The complexity for antecedent generation corresponds to the time added for generating the antecedents during model training compared to the time required for training the base deep model f. Here, N is the number of training samples, and C is the time complexity for computing the consequent of each antecedent. As shown in the table, removing the recursive antecedent generator (RE) or the neural consequent estimator (NE) brings an additional linear complexity with the number of feasible antecedents A, which is much larger than A'. For example, in our experiment, setting A' to 10^4 is good enough to train an accurate neural consequent estimator, while the number of all possible antecedents is $A = 6.25 \times 10^{12}$. Here, we do not include the analysis for sampling A' rules before training the consequent estimator. See Appendix B.4 for more details.

Table 1: Time complexity analysis. -RG and -NE denote our method without recursive antecedent generation and neural consequent estimator.

	Consequent Estimator	Antecedent Generator
SELOR	$O(A'C + A'L^2)$	$O(N\|\mathcal{O}\|L + NL^2)$
-RG	$O(A'C + A'L^2)$	$O(NA + NL^2)$
-RG-NE	$O(AC)$	$O(NA)$

3 Experiment

3.1 Experimental Settings

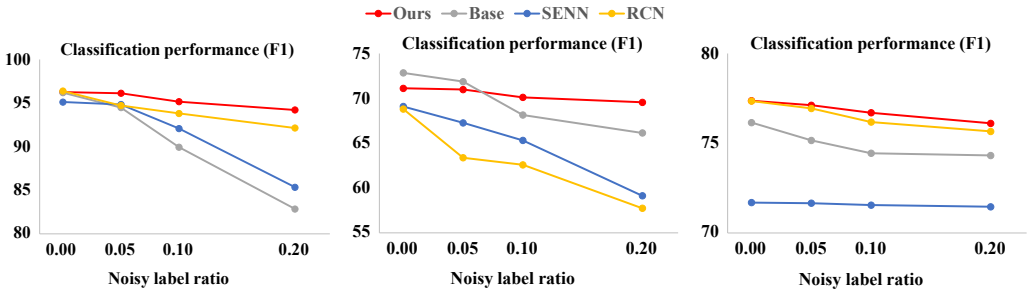
Datasets. We conduct experiments on three datasets. The first two are textual, and the third is tabular. **Yelp** classifies reviews of local businesses into positive or negative sentiment [44], and **Clickbait News Detection** from Kaggle labels whether a news article is a clickbait [45]. **Adult** from the UCI machine learning repository [46], is an imbalanced tabular dataset that provides labels about whether the annual income of an adult is more than \$50K/yr or not. For Yelp, we use a down-sampled subset (10%) for training, as per existing work [39]. More details about the datasets are in Appendix C.1.

Baselines. We compare our model to four baselines. Two self-explainable models, **SENN** [11] and **RCN** [39], are compared in accuracy, robustness, explainability, and efficiency. Two post-hoc explainable methods, **LIME** [1]and **Anchor** [14], are compared in explainability and efficiency.

Implementation details. To match with baselines, we use the AND operation by default in explanations. The impact of using other logical connectives is presented in Appendix C.5.3. The atoms, or interpretable features, are the same as in the majority of baselines, i.e., words for the textual dataset (e.g., "*amazing*") and categorical features for the tabular data (e.g.,"*age*<28"). More implementation details, e.g., selection of atom candidates, are in Appendix C.2.

Table 2: Comparison of classification performance measured in AUC. The average results from five runs are shown. Base refers to the performance of unexplainable backbones (BERT, RoBERTa, DNN). The best results among self-explaining models are marked in **bold**, and the highlighted cells indicate a similar or better result compared with the unexplainable backbone (Base). The numbers in subscript indicates the standard error of the result.

	Yelp		Clickbait		Adult	Average
	BERT	RoBERTa	BERT	RoBERTa	DNN	
Base	97.39 $_{0.0659}$	97.90 $_{0.0577}$	62.27 $_{1.0400}$	63.72 $_{0.8722}$	68.62 $_{0.2317}$	77.98
SENN	96.00 $_{0.1087}$	96.97 $_{0.0841}$	55.64 $_{1.0118}$	57.93 $_{0.7779}$	67.39 $_{0.0854}$	63.20
RCN	**97.31** $_{0.0274}$	**98.03** $_{0.0086}$	59.91 $_{0.2024}$	59.37 $_{0.2259}$	**70.06** $_{0.0411}$	76.94
SELOR	**97.28** $_{0.0335}$	**97.78** $_{0.0833}$	**60.31** $_{0.8498}$	**64.14** $_{0.5906}$	**70.36** $_{0.0892}$	**77.97**



(a) Yelp (b) Clickbait (c) Adult

Figure 2: Experimental results on robustness under different ratios of noisy labels.

3.2 Classification Performance and Robustness

Classification performance. Table 2 shows the classification performance of SELOR and baselines. Here, we evaluate the PR AUC instead of the ROC AUC because the latter is less suitable for imbalanced datasets [47]. BERT and RoBERTa are used as the backbone networks in the NLP datasets, while 3-Layer DNN is used for the tabular dataset. The base method is the vanilla backbone network that does not support explainability (Appendix C.2). The prediction performance of post-hoc methods, LIME and Anchor, is the same as the base model as they utilize the trained model without any extra optimization. Comparison with a fully-transparent model is presented in Appendix C.5.1. Our method achieves comparable average performance with the unexplainable base model and outperforms other self-explaining models by 1.9%. Moreover, our method achieves the best or comparable results on various datasets against backbone models, demonstrating the expressive power of logic rules for high prediction performance. RCN cannot perform as well on challenging textual datasets like Clickbait because it computes soft attention over a predefined rule set. This indicates that (potentially biased) predefined rule sets will limit the model's capability.

Robustness to noisy labels. Following the literature [48, 49], we assess the robustness of SELOR against randomly corrupted labels. We hypothesize that the effect of the noisy label can be alleviated by our consequent estimation term $p(y|\boldsymbol{\alpha})$, where model decisions are verified by testing the logic rule over the entire dataset. For experiments, symmetric noise is introduced by randomly flipping the labels for a subset of the training data. Fig. 2 shows the performance results over Yelp, Clickbait, and Adult datasets with multiple levels of noise ratio from 5% to 20%. Our model outperforms other models across every dataset and every noise ratio. The improvement is substantial even with a high noise ratio (i.e., 20%). For a noise ratio above 10%, our method consistently performs better than the unexplainable base models (2.4% to 13.7%).

Sensitivity analysis. We show that the prediction performance of SELOR is stable under different hyper-parameter settings in Appendix C.5.4.

3.3 Explainability

User study on human precision. To evaluate human precision, we recruited nine native English speakers through a vendor company [50]. Each participant was provided with randomly selected 50 Yelp reviews and 50 Adult samples. Five explanations obtained from different methods were

Table 3: User study results on human precision. We show the average (Avg.) and inter-participant agreement (Agr.) on the percentage of explanations that are considered good (a, b) or best (c, d). One star (*) means p-value is less than 0.05. Best results are highlighted in **bold**.

(a) Percentage of good (Yelp)

	Avg.	Agr.	P-value
Lime	89.8	84.4	8.68 E-04*
Anchor	84.4	87.7	1.12 E-07*
SENN	34.4	72.3	1.40 E-51*
RCN	64.0	77.6	7.26 E-13*
SELOR	**94.4**	93.9	-

(b) Percentage of good (Adult)

	Avg.	Agr.	P-value
Lime	42.7	57.1	6.09 E-54*
Anchor	52.7	59.9	5.56 E-18*
SENN	46.0	51.5	1.18 E-41*
RCN	60.9	53.2	2.83 E-27*
SELOR	**90.7**	85.7	-

(c) Percentage of best (Yelp)

	Avg.	Agr.	P-value
Lime	34.2	67.6	8.87 E-03*
Anchor	18.0	83.6	5.63 E-18*
SENN	2.4	96.3	6.84 E-40*
RCN	2.0	96.3	6.84 E-40*
SELOR	**46.7**	64.8	-

(d) Percentage of best (Adult)

	Avg.	Agr.	P-value
Lime	1.3	96.7	1.72 E-64*
Anchor	13.8	82.9	1.23 E-35*
SENN	9.6	83.3	2.30 E-36*
RCN	10.2	82.9	1.23 E-35*
SELOR	**65.1**	58.4	-

provided for each sample, and the participants reviewed whether the explanations offered reasonable rationales. Participants provided two labels for each explanation, indicating whether it was good or if it was the best one. A **good** explanation should naturally lead to the prediction, but it can contain noisy features. For example, "*amazing*, *are*" is a good explanation for positive sentiment. The **best** explanation is the one that contains the most important and least noisy features. The participants were allowed to choose multiple best explanations only if the chosen ones were the same. For a fair comparison, we show explanations in the same form: a list of features each method considers important for prediction. Example explanations generated by our method and the baselines are given in Fig. 3. Note that ˜ in RCN means negation (NOT). More details about explanation generation, labeling guidelines, and participants' results are given in Appendix C.4.

Table 3 shows that SELOR marks the highest percentage of good explanations, with an average ratio of 94.4% on Yelp and 90.7% on Adult. Our method is also most frequently chosen as the best explanation. All the results are statistically significant according to the p-value from the t-test. Although logic rules are promising, choosing from a small set of predefined rules with attention may not be good enough (RCN) due to the potential bias in the rule set. For example, RCN uses rules extracted with traditional machine learning methods that meet the global data distribution but frequently fail to adequately represent each sample, particularly on datasets with many features (Yelp). As a result, RCN is rarely chosen as the best explanation, especially for Yelp text data. Post-hoc methods also tend to offer good human precision. The best ratio of LIME and Anchor, however, is substantially lower than ours, indicating that the base model may rely on more noisy features for prediction. In contrast, our method can verify its decision by testing the logic rule across the entire dataset.

Case study on model debugging and refinement. What useful insights can SELOR provide on performance? In a study of 20,000 sampled Yelp reviews, we clustered the generated explanations into 10 clusters by applying K-Means on the antecedent embeddings. Table 4 shows five clusters with the lowest training accuracy to illustrate potential reasons for bad performance (see highlighted text). Here, NULL represents an empty atom when the model generated explanations that were shorter than the predefined length L.

...The staff was amazing with him and made him feel so comfortable that he actually sat through an entire cleaning without being upset once...The dentist and the assistants were made to work with children and that made it all very relaxing and enjoyable for him.

LIME	amazing, comfortable, enjoyable, more
Anchor	amazing, so
SENN	about
RCN	amazing, ˜went, ˜seeing, ˜did
Ours	amazing, relaxing, comfortable, enjoyable

Adult		**LIME**	**Anchor**	**SENN**	**RCN**	**Ours**
age	*27*				<32.5	<28
education	*HS-grad*	✓				
educational-num	*9*				<12.5	<10
marital-status	*Separated*		✓	✓	✓	
occupation	*Craft-repair*			✓		✓
relationship	*Own-child*	✓			✓	
race	*White*					
capital-gain	*0*	✓		✓		
capital-loss	*0*	✓		✓		

Figure 3: Example explanations produced by five methods on Yelp (left) and Adult (right).

Table 4: Case study on Yelp. We cluster the explanations for the training samples and show the five clusters with the lowest training accuracy. Num is the number of explanations in the cluster, and Len is the average text length of the reviews. Potential reasons for bad performance are marked in brown.

ID	Acc	Label	Num	Len	Atoms in the explanations (ordered by frequency)
1	99.2	99.1% Neg	1,763 8.82%	643	not(290) bad(233) no(185) mediocre(153) bland(149) never(123) again(122) worst(119) ok(115) disappointing(115) terrible(107)
2	99.2	99.2% Pos	2,730 13.7%	584	great(667) delicious(508) best(330) love(294) fresh(285) tasty(255) definitely(254) friendly(232) perfect(199) amazing(198) favorite(194)
3	98.6	98.6% Pos	2,686 13.4%	548	great(793) friendly(329) always(315) best(304) love(295) fun(223) definitely(218) helpful(163) awesome(159) amazing(136) vegas(117)
4	93.2	57.8% Pos	848 4.24%	119	NULL(1738) great(133) not(67) best(39) service(39) love(34) friendly(29) good(26) awesome(22) fast(22) overpriced(20)
5	83.9	58.1% Neg	62 0.31%	682	NULL(24) nicht(13) un(11) eine(9) service(8) pas(8) der(8) die(7) und(7) um(6) den(5) pour(5) de(5) das(4) prix(4) je(4) zu(3) im(3)

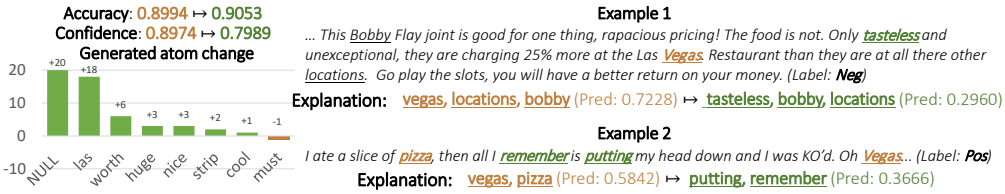


Figure 4: Steering the model without re-training. SELOR allows users to exclude noisy features from explanations during testing, which may simultaneously improve the explanation quality and prediction accuracy. This figure shows the performance before and after removing "*vegas*".

We make the following observations. First, low training accuracy in **cluster 5** is due to non-English reviews, which accounted for 0.31% and led to underfitting. Second, performance degradation also happens when the model does not have enough evidence. For example, reviews in **cluster 4** were short (average length of 119 words) and contained an overwhelming number of NULL atoms (on average 2 per explanation). Third, **cluster 3** contained 13.4% samples that have positive sentiment, and its training accuracy (98.6%) is much larger than cluster 4. However, the cluster also included "*vegas*" in the explanation, which does not seem directly related to sentiment classification. Fourth, **clusters 1 and 2** have reasonably generated atoms, which is consistent with their good training accuracy (99.2%).

SELOR allows us to steer the model directly. For example, after identifying the potentially noisy feature "*vegas*", we can prevent the model from including the term by removing it from the candidate atom list $\mathcal{C}_i$. This type of refinement can be easily achieved during testing, unlike the efforts-taking dataset calibration or model retraining. Fig. 4 shows the performance change of the 169 samples that previously included "*vegas*" in their explanations. The histogram shows the atoms that are more frequently generated after removing "*vegas*". The model sometimes relies on similar atoms such as "*las*" or does not find a good candidate (e.g., choosing NULL), which may lead to decreased confidence. However, the probability of including more meaningful atoms also increases (e.g., "*worth*" in the histogram, and "*tasteless*" in Example 1). One may also verify assumptions by checking the samples whose prediction score changes. For example, after removing "*vegas*", the model can no longer predict Example 2 correctly. In Example 2, there is no strong indication of sentiment, and "*vegas*" may be the most helpful feature. This contradicts our previous assumption that "*vegas*" is not critical for sentiment classification. Instead it can provide new insights and guidance for further improvement (e.g., punishing "*vegas*" with a soft prior instead of directly removing it).

Explanation stability and sensitivity analysis. We discuss the stability of our explanations in Appendix C.6. Our quantitative experiment demonstrates that the explanations generated in different runs are consistent. We also present a case study in that SELOR gives similar explanations for similar inputs. Moreover, we conduct user studies to show that the human precision of the explanations is good for different hyper-parameter settings (Appendix C.5.4).

3.4 Efficiency

Table 5 shows that post-hoc explanation methods like LIME and Anchor require a longer time to generate an explanation. RCN has the largest complexity among the self-explaining methods since

Table 5: Time costs in seconds on Yelp (BERT) and Adult.

	Consequent estimator training	Deep model training (1 epoch)				Explanation generation (1 sample)				
		Base	SENN	RCN	SELOR	LIME	Anchor	SENN	RCN	SELOR
Yelp	2041.4	571.9	224.5	503.7	665.6	55.0	2854.2	0.037	0.071	0.055
Adult	1502.8	12.8	9.1	953.9	98.2	2.5	1.18	0.02	0.17	0.015

it enumerates all possible rules and combines them with soft attention. To alleviate this problem, RCN uses a predefined rule set; hence, its efficiency becomes dependent on the size and quality of the rule set. In contrast, SELOR is trained within acceptable time even for large solution space, and humans only need to define the types of atoms and logical connectives. Our model generated each explanation with a linear complexity with length L, while RCN goes over all possible rules and has exponential complexity with L. Our method required additional time for the neural consequent estimator, taking 35 minutes on Yelp and 25 minutes on Adult. This step is only required once for each dataset and hence is acceptable. The consequent estimator can also be reused.

4 Conclusion and Future Work

This work presented a new framework, SELOR, which incorporates self-explanatory capabilities into a deep model to maintain high prediction performance while also providing high human precision by explaining logic rules. Our method does not require predefined rule sets and can be learned in a differentiable way. Extensive tests involving human evaluation show that our method achieves high prediction performance and human precision while being resistant to noisy labels. Although our method brings multiple advantages, there are still multiple aspects that require more careful study in the future:

Stability. A desirable property for self-explaining models is stability, which requires that similar inputs lead to similar explanations. Unlike SENN [11], which proposes a robustness loss to ensure stable explanations against adversarial inputs, SELOR does not employ such a constraint and cannot guarantee the stability of explanations for inputs with similar raw features. However, our framework theoretically ensures stability is modeled in the selected feature space (see Appendix B.6 for more details), which is partially evaluated by a case study in Appendix C.6.

Applicability. While we explored text and tabular data, our model is applicable to other data types like images and graphs. We can treat a cluster of images or superpixels as an atom [11] or extract atoms with CAV (Concept Activation Vector), a feature that indicates the concept of humans (e.g., striped, red) [51]. End-to-end feature learning is possible in our framework if the number of candidate atoms is small (e.g., around 100 object classes or concepts [35]).

Coverage of the rules. We cannot explicitly model higher-level properties of atoms (e.g., learn that "*awesome*" is a *positive sentiment word* and make a rule based on *positive sentiment word*) since we do not directly consider predicates. Because our rules are constructed with bottom-level atoms (e.g., specific words), they usually have low coverage (e.g., rule "*awesome* AND *tasty*" => *positive sentiment* only covers 0.37% of the input instances). Extending to **first-order logic** may naturally solve this problem, which is an exciting future direction.

Level of insight. SELOR can only find rules constructed with bottom-level atoms instead of summarizing important high-level patterns. If upgraded to first-order logic, the model may directly find high-level patterns such as "*a negation word* AND *a positive sentiment word*" => *negative sentiment*, instead of listing many specific rules such as "*not great*" => *negative sentiment* and "*no good*" => *negative sentiment*. This could save human cognitive budget and improve the reasoning capability of deep models. Moreover, we may automatically compose high-level concepts such as "*strong positive phrase*" and build rules with them. The concept "*strong positive phrase*" may be composed by detecting two consecutive positive sentiment words ("*amazingly comfortable*" and "*perfectly enjoyable*") with predicate invention in [19].

Acknowledgments and Disclosure of Funding

We thank Fangzhao Wu, Sundong Kim, Eunji Lee and the reviewers for their insightful feedback on our work. This research was supported by Microsoft Research Asia, Institute for Basic Science (IBS-R029-C2) in Korea, and the Potential Individuals Global Training Program (2021-0-01696) by the Ministry of Science and ICT in Korea.

References

[1] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?" explaining the predictions of any classifier. In *KDD*, 2016.

[2] Sebastian Thrun. Extracting rules from artificial neural networks with distributed representations. In *NeurIPS*, 1994.

[3] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Nothing else matters: Model-agnostic explanations by identifying prediction invariance. *stat*, 2016.

[4] Tao Lei, Regina Barzilay, and Tommi Jaakkola. Rationalizing neural predictions. In *EMNLP*, 2016.

[5] Dimitrios Alikaniotis, Helen Yannakoudakis, and Marek Rei. Automatic text scoring using neural networks. In *ACL*, 2016.

[6] Hendrik Strobelt, Sebastian Gehrmann, Hanspeter Pfister, and Alexander M Rush. Lstmvis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE TVCG*, 2017.

[7] W James Murdoch, Peter J Liu, and Bin Yu. Beyond word importance: Contextual decomposition to extract interactions from lstms. In *ICLR*, 2018.

[8] Georgina Peake and Jun Wang. Explanation mining: Post hoc interpretability of latent factor models for recommendation systems. In *KDD*, 2018.

[9] Jian Liang, Bing Bai, Yuren Cao, Kun Bai, and Fei Wang. Adversarial infidelity learning for model interpretation. In *KDD*, 2020.

[10] Jingyue Gao, Xiting Wang, Yasha Wang, Yulan Yan, and Xing Xie. Learning groupwise explanations for black-box models. In *IJCAI*, 2021.

[11] David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. In *NeurIPS*, 2018.

[12] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 2019.

[13] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *NeurIPS*, 2017.

[14] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *AAAI*, 2018.

[15] Chaoyu Guan, Xiting Wang, Quanshi Zhang, Runjin Chen, Di He, and Xing Xie. Towards a deep and unified understanding of deep neural models in nlp. In *ICML*, 2019.

[16] Sungsoo Ray Hong, Jessica Hullman, and Enrico Bertini. Human factors in model interpretability: Industry practices, challenges, and needs. *PACM HCI*, 2020.

[17] Benjamin Letham, Cynthia Rudin, Tyler H McCormick, and David Madigan. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *AOAS*, 2015.

[18] Hongyu Yang, Cynthia Rudin, and Margo Seltzer. Scalable bayesian rule lists. In *ICML*, 2017.

[19] Richard Evans and Edward Grefenstette. Learning explanatory rules from noisy data. *Journal of Artificial Intelligence Research*, 2018.

[20] Elaine Angelino, Nicholas Larus-Stone, Daniel Alabi, Margo Seltzer, and Cynthia Rudin. Learning certifiably optimal rule lists. In *KDD*, 2017.

[21] Yao Ming, Panpan Xu, Huamin Qu, and Liu Ren. Interpretable and steerable sequence learning via prototypes. In *KDD*, 2019.

[22] Zhongxia Chen, Xiting Wang, Xing Xie, Mehul Parsana, Akshay Soni, Xiang Ao, and Enhong Chen. Towards explainable conversational recommendation. In *IJCAI*, 2020.

[23] Todd Kulesza, Margaret Burnett, Weng-Keen Wong, and Simone Stumpf. Principles of explanatory debugging to personalize interactive machine learning. In *IUI*, 2015.

[24] Patrick Schramowski, Wolfgang Stammer, Stefano Teso, Anna Brugger, Franziska Herbert, Xiaoting Shao, Hans-Georg Luigs, Anne-Katrin Mahlein, and Kristian Kersting. Making deep neural networks right for the right scientific reasons by interacting with their explanations. *Nature Machine Intelligence*, 2020.

[25] Piyawat Lertvittayakumjorn, Lucia Specia, and Francesca Toni. Find: Human-in-the-loop debugging deep text classifiers. In *EMNLP*, 2020.

[26] Gabriele Ciravegna, Francesco Giannini, Marco Gori, Marco Maggini, and Stefano Melacci. Human-driven fol explanations of deep learning. In *IJCAI*, 2021.

[27] Wolfgang Stammer, Patrick Schramowski, and Kristian Kersting. Right for the right concept: Revising neuro-symbolic concepts by interacting with their explanations. In *CVPR*, 2021.

[28] Andrea Bontempelli, Fausto Giunchiglia, Andrea Passerini, and Stefano Teso. Toward a unified framework for debugging gray-box models. *arXiv preprint arXiv:2109.11160*, 2021.

[29] Luc De Raedt, Sebastijan Dumancic, Robin Manhaeve, and Giuseppe Marra. From statistical relational to neuro-symbolic artificial intelligence. In *IJCAI*, 2020.

[30] Lazar Valkov, Dipak Chaudhari, Akash Srivastava, Charles Sutton, and Swarat Chaudhuri. Houdini: lifelong learning as program synthesis. In *NeurIPS*, 2018.

[31] Kevin Ellis, Lucas Morales, Mathias Sablé-Meyer, Armando Solar-Lezama, and Josh Tenenbaum. Learning libraries of subroutines for neurally-guided bayesian program induction. In *NeurIPS*, 2018.

[32] Ashwin Kalyan, Abhishek Mohta, Oleksandr Polozov, Dhruv Batra, Prateek Jain, and Sumit Gulwani. Neural-guided deductive search for real-time program synthesis from examples. In *ICLR*, 2018.

[33] Kevin Ellis, Daniel Ritchie, Armando Solar-Lezama, and Josh Tenenbaum. Learning to infer graphics programs from hand-drawn images. In *NeurIPS*, 2018.

[34] Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B Tenenbaum, and Jiajun Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In *ICLR*, 2019.

[35] Dongran Yu, Bo Yang, Qianhao Wei, Anchen Li, and Shirui Pan. A probabilistic graphical model based on neural-symbolic reasoning for visual relationship detection. In *CVPR*, 2022.

[36] Xiting Wang, Kunpeng Liu, Dongjie Wang, Le Wu, Yanjie Fu, and Xing Xie. Multi-level recommendation reasoning over knowledge graphs with reinforcement learning. In *WebConf*, 2022.

[37] Fan Yang, Zhilin Yang, and William W Cohen. Differentiable learning of logical rules for knowledge base reasoning. In *NeurIPS*, 2017.

[38] Kangzhi Zhao, Xiting Wang, Yuren Zhang, Li Zhao, Zheng Liu, Chunxiao Xing, and Xing Xie. Leveraging demonstrations for reinforcement recommendation reasoning over knowledge graphs. In *SIGIR*, 2020.

[39] Yuzuru Okajima and Kunihiko Sadamasa. Deep neural networks constrained by decision rules. In *AAAI*, 2019.

[40] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *stat*, 2017.

[41] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, 2018.

[42] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *SSST*, 2014.

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[44] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *NeurIPS*, 2015.

[45] Open Data Science (`ODS.ai`). Kaggle clickbait news detection. `https://www.kaggle.com/c/clickbait-news-detection`, 2020.

[46] Dheeru Dua and Casey Graf. UCI machine learning repository. `http://archive.ics.uci.edu/ml`, 2017.

[47] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3):e0118432, 2015.

[48] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *ICLR*, 2019.

[49] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. Mixup: Beyond empirical risk minimization. In *ICLR*, 2018.

[50] Speechocean. `https://en.speechocean.com/`.

[51] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *ICML*, 2018.

[52] Eyal Peer, David Rothschild, Andrew Gordon, Zak Evernden, and Ekaterina Damer. Data quality of platforms and panels for online behavioral research. *Behavior Research Methods*, 54(4):1643–1662, 2022.

[53] Beth Sagar-Fenton and Lizzy McNeill. How many words do you need to speak a language. *BBC*, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Sec. 4.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Sec. 4.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Sec. 2.1, Sec. 2.2, Sec. 2.3 and Appendix B.3.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Section 2.2, Section 2.4 and Appendix B.3
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Codes are released at Github (*https://github.com/archon159/SELOR*).
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix C.1 and Appendix C.2.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Table 2 and Table 9.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix C.2.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Sec. 3.1 and Appendix C.2.
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] See checklist 3-(a).
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [Yes] See Sec. 3.3, Appendix A, Appendix C.4 and additionally attached guideline files and screenshots.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [Yes] See Appendix A and Appendix C.4.

Appendix

A Supplement for Section 1 (Introduction)

Here, we present details of the user study in Fig. 1b. The figure shows that logic rule explanations achieve higher human precision than linear-regression-based explanations with local stability, while providing a confidence score that correlates with human precision.

We used a vendor company to recruit three native English speakers for the user study (Sec. 3.3). User studies can be performed by 1) hiring a large number of labelers from platforms like Prolific and AMT or 2) hiring a limited number of experienced annotators from a labeling company. While platforms like Prolific make it easy to find many labelers, they are known to be better suited for cognitively simple tasks and may suffer from errors [52]. Our task is challenging for ordinary labelers, as we require them to carefully reason about which features of an adult are useful for predicting his or her income (the Adult dataset) and compare multiple similar explanations. Thus, we validated the model with more experienced annotators hired through a labeling company. To ensure the labelers have an adequate understanding of the task, we provided them with detailed guidelines and examined their initial labels with feedback when a misunderstanding is detected. Such a close interaction would not be possible in crowdsourcing platforms, which may lead to errors and unreliable results.

Each participant was provided 1,000 and 500 randomly selected explanations from SELOR and SENN, respectively. For each explanation, we test whether it can naturally lead to the model prediction according to human perception. Participants were asked to provide 1) the class label for the explanation and 2) how confident they were in their decision by using a 5-point likert scale (HC, i.e., human confidence). For example, given an explanation "*awesome*, *tasty*", the participant will give the label *positive sentiment* and a high confidence score "5" out of 5. When labels were the same to model predictions, human precision was high. We sampled explanations so that their confidence score from models (MC, i.e., model confidence) was evenly distributed and examined how explanation quality varies with the confidence score. Fig. 1b shows how human precision changes with different levels of model confidence. As shown in the figure, logic rule explanations achieve higher human precision than the linear-regression-based explanations, and the model confidence shows a strong correlation with human precision. Here, human precision is the F1-score of machine prediction for give logic rules using human prediction as the ground-truth labels. Table 6 provides more detailed information about our user study. The logic rule with a higher MC level tends to have higher agreement and HC. Also, the logic rule shows better human precision at most MC levels.

User instruction and labeling detail. We describe the instructions given to participants in the attached guideline file (*Labeling_Guidelines_User_Study_Figure1b.pdf*) with detailed description of the task and labeling examples. Participants received an Excel file containing blank labels, which they were instructed to fill out and return. The snapshot of the Excel file is also attached as a separate file (*Screenshot_User_Study_Figure1b.PNG*). Each participant was paid 22.5$ per hour and the total budget we spent was 937.5$ for this task.

Table 6: User study results for human precision on logic rule- and linear-regression-based explanations. HC denotes the average human confidence while MC denotes the machine confidence. Avg. denotes the average number of sentiment agreement, human confidence, and human precision of all data points. (Lv 1: 0.0 $\sim$ 0.2, Lv 2: 0.2 $\sim$ 0.4, Lv 3: 0.4 $\sim$ 0.6, Lv 4: 0.6 $\sim$ 0.8, Lv 5: 0.8 $\sim$ 1.0)

(a) Logic rule

	Sentiment Agreement	Avg HC	Human Precision
MC Lv 1	82.67	2.72	52.65
MC Lv 2	86.00	2.88	53.53
MC Lv 3	84.00	3.31	76.19
MC Lv 4	92.67	3.85	89.38
MC Lv 5	95.00	4.07	90.41
Avg.	88.07	3.36	73.32

(b) Linear-regression-based explanation

	Sentiment Agreement	Avg HC	Human Precision
MC Lv 1	78.79	3.63	47.71
MC Lv 2	81.56	3.55	48.69
MC Lv 3	75.95	3.62	52.83
MC Lv 4	79.12	3.56	56.82
MC Lv 5	84.51	3.78	66.17
Avg.	79.96	3.63	54.46

B Supplement for Section 2 (Deep Logic Rule Reasoning)

B.1 Symbols

Table 7 summarizes the symbols used in this paper.

Table 7: The meaning and detailed explanation of each symbol used in the paper.

	Meaning	Detailed Explanation
$\mathbf{x}$	Input sample	Any type of data (e.g. text, tabular)
$\boldsymbol{\alpha}$	Antecedent	Condition to apply the rule
b	Human belief	Common sense that a human believes when they make a decision
y	Consequent	Model's prediction output for the given antecedent
o_i	Atom or logical connective	Atom is the smallest unit of explanation
		Logical connective combines atoms
$\mathbf{o}_i$	Embedding of o_i	Initialized as the average embedding of all training samples that satisfy the atom
$\mathcal{O}$	Set of atoms	
$\mathcal{C}_i$	Set of candidates for o_i	Every candidate should satisfy both global and local constraints (Sec. 2.5)
L	Length of an antecedent	Number of atoms and logical connectives included in an antecedent
N	Number of training data	
$n_{\boldsymbol{\alpha}}$	See detailed explanation	Number of data samples in training data that satisfies the antecedent $\boldsymbol{\alpha}$
$n_{\boldsymbol{\alpha},y}$	See detailed explanation	Number of data samples in training data that satisfies the antecedent $\boldsymbol{\alpha}$ and has the consequent y
$\mathcal{Y}_{\boldsymbol{\alpha}}$	See detailed explanation	Data samples of class y in training data that satisfies the antecedent $\boldsymbol{\alpha}$
$\boldsymbol{\alpha}^{(s)}$	s-th sample of $\boldsymbol{\alpha}$	s-th sampled antecedent in deep antecedent generation
S	Total number of $\boldsymbol{\alpha}^{(s)}$	Set as S = 1 by default
$\Omega(\boldsymbol{\alpha})$	Required number of $\boldsymbol{\alpha}$	The number of explanation required to explain given input
$\mathbf{h}_i$	Hidden state of encoder	The encoder can be any neural sequence encoder such as GRU or Transformer
C	See detailed explanation	Time complexity for computing the consequent of each antecedent
A	See detailed explanation	Number of all feasible antecedents. Usually exponentially increase with $\|\mathcal{O}\|$ and L (i.e. $\|\mathcal{O}\|^L$)
A'	See detailed explanation	Number of sampled antecedents for training of neural consequent estimator

B.2 Extension to Regression Tasks

Although we mainly focused on classification tasks, SELOR can be applied to regression tasks after a small modification. For regression tasks, we change the modeling of neural consequent estimation from a categorical to a direct prediction. Our neural consequent estimator for regression predicts the value y' instead of $p(y|\boldsymbol{\alpha})$ and coverage c_α. Then, we maximize $||y' - y||_2$.

B.3 Probability Decomposition

Here we give the proof for Eq. (1):

$$
\begin{aligned}
p(y|\mathbf{x}, b) &= \sum_{\boldsymbol{\alpha}} p(y|\boldsymbol{\alpha}) p(\boldsymbol{\alpha}|\mathbf{x}, b) \\
&= \sum_{\boldsymbol{\alpha}} p(y|\boldsymbol{\alpha}) \frac{p(\boldsymbol{\alpha}, \mathbf{x}, b)}{p(\mathbf{x}, b)} \\
&= \sum_{\boldsymbol{\alpha}} p(y|\boldsymbol{\alpha}) \cdot \frac{p(\boldsymbol{\alpha}, \mathbf{x}, b)}{p(x, b)} \cdot \frac{p(b|\boldsymbol{\alpha}) p(\boldsymbol{\alpha})}{p(\boldsymbol{\alpha}, b)} \cdot \frac{p(\boldsymbol{\alpha}|\mathbf{x}) p(x)}{p(\boldsymbol{\alpha}, \mathbf{x})} && (9) \\
&= \sum_{\boldsymbol{\alpha}} p(y|\boldsymbol{\alpha}) \cdot p(b|\boldsymbol{\alpha}) \cdot p(\boldsymbol{\alpha}|\mathbf{x}) \cdot \frac{1}{p(b)} \cdot \frac{p(\mathbf{x}) p(b)}{p(\mathbf{x}, b)} \cdot \frac{p(\boldsymbol{\alpha}) p(\boldsymbol{\alpha}, \mathbf{x}, b)}{p(\boldsymbol{\alpha}, \mathbf{x}) p(\boldsymbol{\alpha}, b)} && (10) \\
&= \sum_{\boldsymbol{\alpha}} p(y|\boldsymbol{\alpha}) \cdot p(b|\boldsymbol{\alpha}) \cdot p(\boldsymbol{\alpha}|\mathbf{x}) \cdot \frac{1}{p(b)} \cdot \frac{p(\mathbf{x}) p(b)}{p(\mathbf{x}, b)} \cdot \frac{p(\mathbf{x}, b|\boldsymbol{\alpha})}{p(\mathbf{x}|\boldsymbol{\alpha}) p(b|\boldsymbol{\alpha})} && (11) \\
&\propto \sum_{\boldsymbol{\alpha}} p(y|\boldsymbol{\alpha}) \cdot p(b|\boldsymbol{\alpha}) \cdot p(\boldsymbol{\alpha}|\mathbf{x}) && (12)
\end{aligned}
$$

There are two assumptions to hold Eq. (12).

Assumption A. For $p(y|x, b) = \sum_{\boldsymbol{\alpha}} p(y|\boldsymbol{\alpha}) p(\boldsymbol{\alpha}|\mathbf{x}, b)$, we assume that $p(y|\boldsymbol{\alpha}) = p(y|\boldsymbol{\alpha}, \mathbf{x}, b)$. This is decomposed into two assumptions: $p(y|\boldsymbol{\alpha}) = p(y|\boldsymbol{\alpha}, \mathbf{x})$ (A1) and $p(y|\boldsymbol{\alpha}, x) = p(y|\boldsymbol{\alpha}, \mathbf{x}, b)$ (A2).

Assumption A1 $p(y|\boldsymbol{\alpha}) = p(y|\boldsymbol{\alpha}, \mathbf{x})$ indicates that explanation $\boldsymbol{\alpha}$ contains all information in input $\mathbf{x}$ that is needed to predict y. This formulation compels the model to pass information from $\mathbf{x}$ to y only via explanations, as opposed to other unexplainable parts. This assumption may limit the prediction performance, but it is essential for $\boldsymbol{\alpha}$ to be a trustable explanation for predicting y. Otherwise, there may be a direct connection between y and $\mathbf{x}$ that is unrelated to the explanation $\boldsymbol{\alpha}$. Thus, $\boldsymbol{\alpha}$ may only explain a small portion of the model behavior (e.g., only explain 1% of the change in y) and differ substantially from the ground-truth explanation of the model behavior.

Assumption A2 $p(y|\boldsymbol{\alpha}, x) = p(y|\boldsymbol{\alpha}, \mathbf{x}, b)$ means that explanation $\boldsymbol{\alpha}$ and input $\mathbf{x}$ contain all of the information in b (human prior preference for explanations) that is needed to predict y. It is intuitive that this assumption holds, as human preference for explanations is unrelated to the current class label.

Assumption B. For $\sum_{\boldsymbol{\alpha}} p(y|\boldsymbol{\alpha}) p(\boldsymbol{\alpha}|\mathbf{x}, b) \propto \sum_{\boldsymbol{\alpha}} p(b|\boldsymbol{\alpha}) p(y|\boldsymbol{\alpha}) p(\boldsymbol{\alpha}|\mathbf{x})$, we assume that $p(\mathbf{x}, b) = p(b) p(\mathbf{x})$ and $p(\mathbf{x}, b|\boldsymbol{\alpha}) = p(b|\boldsymbol{\alpha}) p(\mathbf{x}|\boldsymbol{\alpha})$.

It means that $\mathbf{x}$ and b are independent no matter which $\boldsymbol{\alpha}$ is given. In other words, seeing input sample ($\mathbf{x}$) does not change the belief in our prior preference for explanations (b), no matter which explanations ($\boldsymbol{\alpha}$) are given, i.e., $p(b) = p(b|\mathbf{x})$ and $p(b|\boldsymbol{\alpha}) = p(b|\mathbf{x}, \boldsymbol{\alpha})$. The rationale for this assumption is that human preferences for explanations are usually fixed and unrelated with the input $\mathbf{x}$. Even if this assumption is not satisfied, it will not have a significant effect on the framework. Only the human prior module must be integrated into the antecedent generation module, which changes from $p(\boldsymbol{\alpha}|\mathbf{x})$ to $p(\boldsymbol{\alpha}|\mathbf{x}, b)$.

B.4 Neural Consequent Estimation

The input of the neural consequent estimator is the antecedent embedding, which is obtained by $(\mathbf{o}_1 ..., \mathbf{o}_L)$, where $\mathbf{o}_i$ is the embedding of o_i in $\boldsymbol{\alpha} = (o_1 ..., o_L)$. For each atom, $\mathbf{o}_i$ is initialized as the average embedding of all training samples that satisfy the atom, where the sample embedding can be derived using a pretrained model or f. The embeddings of logical connectives are initialized at random, and can be omitted when there is only one logical connective (e.g., AND). We use the Transformer encoder [43] as the backbone neural network to emphasize the contextual interaction between atoms and logical connectives.

After encoding $\boldsymbol{\alpha}$ with Transformer, an MLP (Multi-Layer Perceptron) layer reduces the representation obtained by mean pooling to a logit. Softmax (multi-class) or sigmoid (two classes) is used to activate the logits to determine the probability for each class $p(y|\boldsymbol{\alpha})$ and the coverage c_α of the antecedent $\boldsymbol{\alpha}$, which is converted to the number of observations in the training dataset with

$n_\alpha = c_\alpha N$. The time complexity of deep logic reasoning is significantly reduced by neural estimation of the consequence (Sec. 2.6).

The neural consequent estimator is pretrained with $A' = 10,000$ sampled rules for each antecedent length (Total $L \times A'$), then used to train the deep antecedent generator with frozen parameters. The following steps are taken to ensure the generality of the rules used in pretraining. To begin, we create the "true matrix" (*tm*), that has the size $(|\mathcal{O}| \times N)$, which indicates whether each input sample satisfies each atoms. Then, by multiplying *tm* and its transpose, we can create a matrix of size $(|\mathcal{O}| \times |\mathcal{O}|)$ that indicates the number of samples that satisfy 2-length antecedents ($[o_i, o_j], i, j \in \mathcal{O}$).

Then, we obtain the list of 2-length antecedents whose frequency is larger than a threshold (i.e., *min_df*). From the 2-length antecedent list, we sample $k \times A'$ rules while k is a hyper-parameter larger than 1. We set k to be the same with *min_df* in the experiment. With these $k \times A'$ rules, we can make a new true matrix of size $((k \times A') \times N)$ and repeat the steps to obtain the rules whose frequency is larger than *min_df*. This sampling process takes linear time to A' instead of A, which reduces the time complexity. After the whole process, we can obtain $k \times A'$ number of antecedents for each length. Then we randomly choose A' rules for pretraining of consequent estimator maintaining the balance of labels. In practice, the time spent in sampling process was 1144s for Yelp, 402s for Clickbait, and 456s for Adult dataset in our setting. This time can be even reduced with larger *min_df*.

B.5 Differentiable Learning

In Sec. 2.6, we sampled one antecedent from $p(\boldsymbol{\alpha}^{(s)}|x)$. Naive selection (e.g., selecting the maximum value's index) stops the gradient and prevents differential learning of the neural model. This problem is solved by sampling $\boldsymbol{\alpha}$ with the Straight-Through Gumbel-Softmax function, as shown in Eq. (7). For forward propagation, $\boldsymbol{\alpha}^{(s)} = (\alpha_1^{(s)}..., \alpha_L^{(s)})$ is represented by L discrete one-hot vectors. To derive L input embeddings for the neural consequent estimator in Sec. 2.5, each one-hot vector is multiplied by an embedding matrix of atoms and logical connectives. Differentiable Gumbel-Softmax distribution is used to approximate the gradients during backpropagation.

B.6 Theoretical Analysis of Explanation Stability

For linear-regression-based models like SENN [11], the explanations for similar inputs may be entirely different without specific constraints like the robustness loss, because the main optimization goal for SENN is the local prediction accuracy. Without the robustness loss, the model may find a correct prediction locally for a single instance, but being "surely no more interpretable than any deep neural network" (quoted from the SENN paper). However, this is not the case for the logic rule reasoning framework, because the antecedent generator is trained to optimize two globally consistent rewards (Eq. 3 and Sec. 2.6): human's prior belief about which explanation types are good and the explanation confidence that is measured by the global prediction accuracy over the entire training dataset given the explanation (logic rule). Thus, explanations for similar inputs may be different only when:

1. The optimal (most confident and human-preferred) rules for the inputs are different.
2. There are multiple explanations that achieve the exact same reward.
3. The model has not been trained sufficiently to achieve the optimal result.

In situation 1), SELOR removes the heuristic constraint regarding the similarity of explanations, allowing us to identify the optimal explanations for the two inputs. If an instance A is changed to the instance B by substituting "*very disappointing*" with "*disappointing*", then the best explanation may change from "*very disappointing*" in the instance A to "*awful*" in the instance B. Even if the two instances are similar, their optimal explanations may differ. This is plausible as such a change increases the explanation's confidence. In other words, the radius of validity of an explanation corresponds to inputs that have similar optimal rules. For example, explanation "*very disappointing*"⇒*negative sentiment* can generalize to all instances that satisfy the rule and at the same time do not satisfy the more confident rule. When we want to force the explanations of two inputs to be similar, we can also incorporate a constraint that mimics the robustness loss in SENN into the soft human prior. Situation 2) rarely occurs, as our explanation confidence reward is a real number, not a discrete value. In rare cases where this occurs, it is possible to remedy the situation by using the soft human prior. Situation 3) can be avoided by checking the training loss, the classification accuracy, and the explainability.

C Supplement for Section 3 (Experiment)

C.1 Datasets

We use the following three datasets for experiments. Table 8 reports the number of data points for each dataset that we used for training, validation, and testing.

Yelp classifies reviews of local businesses into positive or negative sentiment [44]. For Yelp, we use a down-sampled subset (10%) for training, as per existing work [39]. We split the test dataset and used half of them for the validation dataset.

Clickbait News Detection from Kaggle labels whether a news article is a clickbait [45], and we use the "*news*" and "*clickbait*" classes in the dataset. We split the train data into train and validation.

Adult from the UCI machine learning repository [46] is an imbalanced tabular dataset that provides labels about whether the annual income of an adult is more than \$50K/yr or not. We split the data points into train, validation, and test datasets.

C.2 Implementation Details

Hyperparameter settings. The backbone models for textual data (i.e., BERT, RoBERTa) follow the original setting, and the model for tabular data (i.e., DNN) consists of network with three fully-connected layers with ReLU activation layers (i.e., FC-ReLU-FC-ReLU-FC) with 512 hidden dimensions. We employ GRU [42] as a sequential encoder for deep antecedent generation, and Transformer [43] as a neural model for consequent estimation, respectively. For deep antecedent generation, neural consequent estimation, and other baseline models, we set the hidden dimension $|h|$ as the default BERT and RoBERTa embedding size (i.e., 768) for textual data and 512 for tabular data. For training of SELOR, cross-entropy loss is used for optimization on the probability predicted by the consequent estimator for the antecedents extracted by the antecedent generator. For RCN, we extract a predefined rule set by following the original work [39]. In particular, the predefined rules are decision paths in random forests with 100 estimators and a maximum depth of four. After excluding stopwords, we limit atoms in textual data to only derive from the top-5000 most frequent words. Tabular data uses both categorical and numerical features for atoms while the threshold of numerical features is set to the 25th, 50th, 75th percentiles of data. The length of antecedent L (i.e., the number of atoms from recursive deep antecedent generation) is set to 4. The minimum document frequency is set to 200, and the number of rules for pretraining the neural consequent estimator is set to $10,000$.

We introduce hyper-parameters in training our model and baselines. Note that the same hyper-parameters are used for training baselines, the neural consequent estimator, and the deep antecedent generator for the all datasets. The base backbone network and self-explainable models are trained 10 epochs. The batch size is set to 16, the largest size that can be trained on our GPU. For optimization, we employ Adam optimizer with a learning rate of $1e-5$, and ExponentialLR scheduler with γ 0.95. For the learning rate, the one with the best performance is selected after experiments on $5e-5$, $4e-5$, $3e-5$, $2e-5$, and $1e-5$. For SENN, a set of token embeddings from the pretrained language model (i.e., BERT and RoBERTa) are utilized as inputs and are considered to be interpretable basic concepts for textual data experiments. In the case of tabular data, raw input features are used. We follow the implementation and hyper-parameter settings for training as in the original work [11]. Optimizer or scheduler are also set to be the same as other baselines for a fair comparison. One NVIDIA A100 is used for each experiment.

Details about selecting atom candidates. We ensure that atoms have a consistent form with baselines for fair comparison. In current implementation, we only consider atoms like "*awesome* ≥ 1" for

Table 8: Dataset statistics. The labeling ratio shows whether the data is imbalanced between classes. All data, including training, validation, and test data, is split into the same ratio.

Dataset	# for training	# for validation	# for test	Prediction Labels	Label Ratio
Yelp	56000	19000	19000	Negative, Positive	1 : 1
Clickbait	18330	1312	1312	News, Clickbait	3.9 : 1
Adult	39073	4884	4885	<=50K, >50K	3.2 : 1

Table 9: Comparison of classification performance measured in F1. The average results from five runs are shown. The best results among self-explaining models are marked in **bold**, and the highlighted cells indicate a similar or better result compared with the unexplainable backbone (Base). The numbers in subscript indicates the standard error of the result.

	Yelp		Clickbait		Adult	Average
	BERT	RoBERTa	BERT	RoBERTa	DNN	
Base	96.20 $_{0.0541}$	97.16 $_{0.0672}$	72.84 $_{0.9302}$	74.25 $_{0.7763}$	76.15 $_{0.2522}$	83.32
SENN	95.12 $_{0.1995}$	96.07 $_{0.1180}$	69.09 $_{0.9550}$	70.99 $_{0.5076}$	71.69 $_{0.7681}$	80.59
RCN	**96.38** $_{0.0089}$	**97.36** $_{0.0049}$	68.80 $_{0.1359}$	68.64 $_{0.1467}$	**77.35** $_{0.0309}$	81.77
SELOR	**96.26** $_{0.0445}$	**97.13** $_{0.0642}$	**71.12** $_{0.5479}$	**74.20** $_{0.5009}$	**77.37** $_{0.0541}$	**83.34**

textual datasets. This enables a comparison with explainable models that highlights the words based on their importance weight. For tabular datasets, we choose different strategies based on feature types. For categorical features, whether the instance belongs to a certain category or not becomes an atom. For example, in the Adult dataset, "*marital-status* == *Married*" indicates the person in the given instance is married. For numerical features, we calculate 25th, 50th, 75th percentiles of each feature distribution for the threshold. We use whether the feature of a given sample is larger or smaller than the threshold as atoms to obtain thresholds and use those values to determine the over or under presence of each feature in the given sample. For example, the feature "*age*" of the Adult dataset has thresholds 28, 37, and 48, which lead to atoms like "*age* $\geq$ 28", "*age* < 28", "*age* $\geq$ 37", "*age* < 37", "*age* $\geq$ 48", and "*age* < 48". This form is consistent with the atoms in our baseline RCN [39], which uses random forests for rule creation.

It is possible that different atoms associated with the same feature appear in the same explanation, for example, as in our tabular dataset (e.g., "*age* $\geq$ 37" and "*age* $\geq$ 48" for the feature "*age*"). In such a situation, we remove the redundant atoms after the explanation has been generated (e.g., removing "*age* $\geq$ 37"). Note that the generated atoms will not be conflicted with each other. For example, "*age* $\geq$ 48" and "*age* < 37" will not be generated simultaneously in one explanation, because the condition for generation is that the corresponding instance satisfies both atoms. This is enforced by the local constraint introduced in Sec. 2.5. We find that such a post-processing step of removing redundant atoms is easy to implement and has reasonably good explainability and prediction performance. It is also possible to eliminate redundant atoms during explanation generation. One possible way is to create the atoms so that they do not overlap (e.g., creating "*age* $\geq$ 48", "48 > *age* $\geq$ 37", "37 > *age* $\geq$ 28", "28 > *age*" for feature "*age*"). However, this may make it impossible to flexibly combine different thresholds (e.g., generating "48 > *age* $\geq$ 28"). Another way is to apply a mask to the model so that it assigns zero probability to an already chosen feature or a redundant atom. This can be implemented by carefully setting the local constraint in Sec. 2.5.

C.3 Prediction Performance in F1-score

We also provide the prediction performance of SELOR and other self-explainable baselines in Table 9. The result shows that our method successfully maintains the representation ability of deep learning.

C.4 User Study Details

Explanation generation process. Here, we introduce how we generate the explanations.

- **LIME** is distributed as a Python package, and we use *lime_text* and *lime_tabular* to generate explanations. The number of disturbances is set to 3,000 for textual data. We choose the words that are consistent with the prediction having positive weights as explanation. To reduce the incongruity with other explanations, we hide the score provided by LIME and join chosen the predicates.
- **Anchor** is initialized with an empty set. For every iteration, multiple candidate anchors are produced by extending the current anchor by one additional predicate. Then, the model selects the set of predicates with the highest precision as an anchor while perturbing the other predicates. This process repeats until it satisfies the precision constraint of probability 0.95.
- **SENN** defines the interpretable basis concepts $h(x)$ from the input $\mathbf{x}$, and learns the relevance value $\theta(x)$ which is an interpretable weight in relation to each concept (i.e., $f(x) = \sum_i \theta(x)_i \cdot h(x)_i$). We choose the set of top-k predicates with the highest positive relevance value as an interpretation

Table 10: User study results on human precision. Nine participants P1-P9 were asked to annotate whether an explanation is a reasonable rationale for the prediction. For each compared method, we report the percentage of explanations that are considered good (a, b) or best (c, d). Avg. and Agr. denote the average and inter-participant agreement, respectively. P-values from t-test indicates the statistical significance of the experiment. We mark one star (*) if the p-value is lower than 0.05. Best results are highlighted in **bold**.

(a) Percentage of good (Yelp)

	P1	P2	P3	P4	P5	P6	P7	P8	P9	Avg.	Agr.	P-value
Lime	88	82	**96**	90	92	90	98	88	84	89.8	84.4	8.68 E-04*
Anchor	86	74	92	86	84	84	90	78	86	84.4	87.7	1.12 E-07*
SENN	26	22	18	32	26	30	80	32	44	34.4	72.3	1.40 E-51*
RCN	70	32	6	70	62	74	88	76	**98**	64.0	77.6	7.26 E-13*
SELOR	**90**	**84**	**96**	**98**	**100**	**96**	**100**	**92**	94	**94.4**	93.9	-

(b) Percentage of good (Adult)

	P1	P2	P3	P4	P5	P6	P7	P8	P9	Avg.	Agr.	P-value
Lime	**88**	24	88	26	76	2	6	26	48	42.7	57.1	6.09 E-54*
Anchor	30	38	32	54	30	84	68	94	44	52.7	59.9	5.56 E-18*
SENN	**88**	16	**90**	30	82	4	8	38	58	46.0	51.5	1.18 E-41*
RCN	78	70	86	70	56	4	18	86	80	60.9	53.2	2.83 E-27*
SELOR	84	**88**	**90**	**98**	**84**	**98**	**86**	**100**	**88**	**90.7**	85.7	-

(c) Percentage of best (Yelp)

	P1	P2	P3	P4	P5	P6	P7	P8	P9	Avg.	Agr.	P-value
Lime	30	36	24	40	44	34	14	**48**	38	34.2	67.6	8.87 E-03*
Anchor	24	20	**36**	16	8	12	16	10	20	18.0	83.6	5.63 E-18*
SENN	2	4	4	2	2	2	4	0	2	2.4	96.3	6.84 E-40*
RCN	2	2	2	0	0	0	6	6	**0**	2.0	96.3	6.84 E-40*
SELOR	**44**	**40**	**36**	**48**	**50**	**54**	**64**	40	**44**	**46.7**	64.8	-

(d) Percentage of best (Adult)

	P1	P2	P3	P4	P5	P6	P7	P8	P9	Avg.	Agr.	P-value
Lime	0	2	0	0	8	0	0	0	2	1.3	96.7	1.72 E-64*
Anchor	22	18	20	16	2	2	16	2	26	13.8	82.9	1.23 E-35*
SENN	6	10	8	8	34	4	2	2	12	9.6	83.3	2.30 E-36*
RCN	10	10	8	12	10	0	6	10	26	10.2	82.9	1.23 E-35*
SELOR	**62**	**60**	**64**	**64**	**46**	**94**	**76**	**86**	**34**	**65.1**	58.4	-

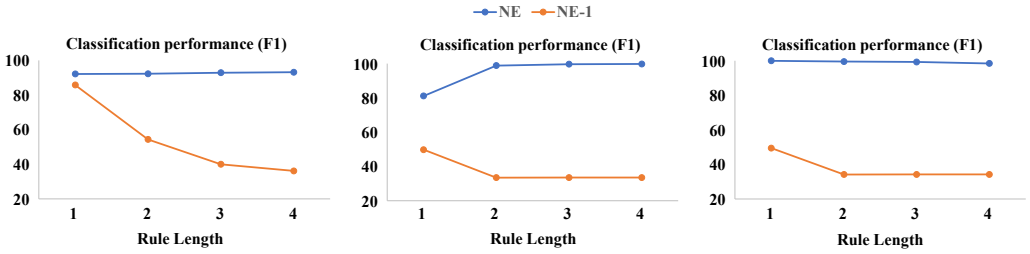
for the given input x. k is set to 5. We remove meaningless words (such as "-", " ") by post-processing. To reduce incongruity with other explanations, we hide the score provided by SENN and join the chosen predicates.

- **RCN** chooses one of the rules from a predefined rule set made from random forest. As the random forest is trained with the bag-of-words of training data, the form of the rule also aligns with the frequency of words.
- **SELOR** recursive deep antecedent generation chooses atoms with the largest weight sequentially. All our atoms are existence of a word (e.g., if "*good*" exists), so a rule becomes the list of words. We join these words to explain the given sample.

User instruction and labeling detail. We provided instructions for participants in the form of the guideline file (*Labeling_Guidelines_User_Study_Table3.pdf*) with detailed description of the task and labeling examples. Participants received an Excel file with empty labels, which they were instructed to fill out and return. The snapshot of the Excel file is also attached as a separate file (*Screenshot_User_Study_Table3_1.PNG*, *Screenshot_User_Study_Table3_2.PNG*). We originally allowed multiple choices as best explanations, but labelers found it unclear how to decide two explanations are equally good. As this guideline led to confusion and further lower agreement among labelers, we updated the guideline to allow only one best explanation. We conducted the user study twice. During the first survey, we hired three participants. For Yelp dataset, each participant was paid $22.5\$$ per hour with the total budget $45\$$. For Adult dataset, each participant was paid $7.5\$$ per hour

Table 11: Performance comparison of SELOR and a fully transparent model, Random Forest. The backbone model of textual dataset for SELOR is RoBERTa.

	Yelp		Clickbait		Adult	
	F1	AUC	F1	AUC	F1	AUC
Random Forest	73.03	80.40	44.29	60.25	65.60	66.15
SELOR	97.13	97.78	74.20	64.14	77.37	70.36



(a) Yelp (b) Clickbait (c) Adult

Figure 5: The prediction performance of consequent estimator with varying length of antecedents. NE-1 denotes the estimator which is only trained with length-1 antecedents.

with total budget 37.5$. At the second survey, we hired six participants, and each participant was paid 7$ per hour for both datasets. The total budget we spent in the second survey was 56$.

Results of all participants. Table 10 provides more detailed result including that of each participant.

Further discussion on user study results. Table 10a shows that participants have a low level of agreement on RCN. This is because people have varying preferences for the logical connective NOT. NOT denotes that the prediction is made due to the absence of a particular feature in the text. One participant (P3) considered most explanations that contained NOT to be noisy because s/he seldomly made decisions based on the absence of a word.

C.5 Additional Experimental Results

We describe additional experimental results to support the prediction performance and explanation quality of SELOR.

C.5.1 Comparison with Fully Transparent Model

Tree-based models are popular explainable models because their decision process is fully transparent. However, fully transparent models such as decision trees and random forests cannot achieve comparable prediction performance to deep models as shown in Table 11.

C.5.2 Effectiveness of Neural Consequent Estimator

The Fig. 5 shows the prediction performance of our neural consequent estimator (NE) for antecedents of varying length. Our consequent estimator shows reasonable performance in most cases. NE-1 is the estimator that is only pretrained with length-1 antecedents and hence cannot learn the relationship among atoms. Its prediction ability dramatically drops for rules longer than 1.

Also, we explore the effect of neural consequent estimator to the overall model performance. Fig. 6 demonstrates that SELOR is not highly sensitive to the number of samples used in pretraining the neural consequent estimator, although it requires a minimum level of prediction ability. Additionally, a larger number of samples are needed for more difficult dataset such as Clickbait.

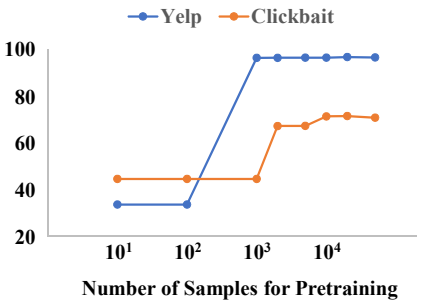


Figure 6: The performance of SELOR with varying number of samples used in pretraining of the neural consequent estimator.

C.5.3 Using Different Logical Connectives

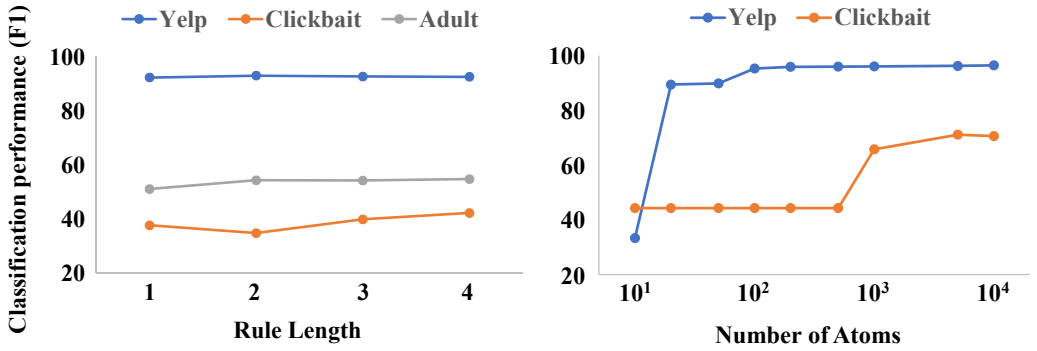
We investigate the performance in terms of F1 of different logical connectives on Yelp using BERT as a base model. First, joining atoms with logical connectives **OR** leads to a prediction performance of 96.21, which is similar to the original model using the **AND** connectives. We also change half of atoms to non-existence rules, which indicates the non-existence of a word (e.g. "NOT *awesome*" means the given instance does not contain the word "*awesome*"). The performance changes to 94.46, and this is natural as the information capacity of non-existence is usually smaller than the existence rules. Additionally, we try **ORDERED AND**, which considers the order of atoms. For example, "*not* BEFORE *happy*" and "*happy* BEFORE *not*" will be treated as different antecedents although they have the same words in atoms. Its performance is 96.93, as the amount of information in the rule increases.

C.5.4 Hyper-Parameter Sensitivity Analysis

We conduct analysis to test sensitivity of two hyper-parameters: antecedent length L and number of atoms $|\mathcal{O}|$.

Impact of hyper-parameters on prediction performance. Fig. 7a shows that SELOR is not sensitive to the length of antecedent although longer antecedents yield better result in general. Fig 7b shows that the number of atoms required for good performance varied by datasets. The more difficult dataset, Clickbait, requires a larger number of atoms to get reasonable performance. However, after certain points, the prediction performance of our method becomes insensitive to number of atoms.

Impact of hyper-parameters on explainability. Table 12 show how human precision of explanations change with the antecedent length. Antecedents of all lengths, including short antecedents with only one atom, offer a certain level of explainability; The average percentage of good for Length 1 antecedent is 79.7%. Meanwhile, longer antecedents tend to improve human precision. This indicates the longer antecedents contain more useful information for decision making as it has more chances to find a good atom, resulting in greater precision. Note that the *length of antecedent* is the maximum length of the antecedent; our method can automatically generate shorter antecedents than the default length by electing the NULL atom. Table 13 shows how human precision of explanations change with the number of candidate atoms. In particular, $1,000$ means that we use the top $1,000$ frequent words as candidate atoms. The explanation quality increases with increasing number of atoms, up to a certain points(i.e., $1,000$ atoms). After this point, there is no statistically significant gain in explainability, demonstrating that SELOR requires a reasonable size of approximately $1,000$ atoms to provide a good explanation. This finding aligns with the observations in [53], which shows that analyzing and explaining text contents such as restaurant reviews and news articles does not require a large vocabulary.



(a) Antecedent Length

(b) Number of Atoms

Figure 7: The predictive performance of our model with varying antecedent lengths and the number of atoms.

Table 12: User study results on human precision with varying antecedent lengths. Participants P1-P6 were asked to annotate whether an explanation is a reasonable rationale for the prediction. For each length, we report the percentage of explanations that are considered good (a) or best (b). Avg. and Agr. denote the average and inter-participant agreement, respectively. P-values from t-test indicates the statistical significance of the experiment. We mark one star (*) if the p-value is lower than 0.05. Best results are highlighted in **bold**.

(a) Percentage of good

	P1	P2	P3	P4	P5	P6	Avg.	Agr.	P-value
Length 1	76	74	76	76	96	80	79.7	87.6	1.31 E-11*
Length 2	92	90	86	94	**100**	84	91.0	91.3	8.93 E-04*
Length 3	96	94	88	92	**100**	84	92.3	89.7	3.73 E-03*
Length 4	**100**	**100**	**90**	**98**	**100**	**86**	**95.6**	91.9	-

(b) Percentage of best

	P1	P2	P3	P4	P5	P6	Avg.	Agr.	P-value
Length 1	8	4	2	6	10	8	6.3	94.3	2.38 E-43*
Length 2	24	26	10	12	14	10	16.0	81.6	6.98 E-23*
Length 3	20	28	18	28	14	26	22.3	81.7	1.58 E-16*
Length 4	**56**	**50**	**68**	**62**	**72**	**58**	**61.0**	66.5	-

Table 13: User study results on human precision with varying number of atoms. Participants P1-P6 were asked to annotate whether an explanation is a reasonable rationale for the prediction. For each length, we report the percentage of explanations that are considered good (a) or best (b). Avg. and Agr. denote the average and inter-participant agreement, respectively. P-values from t-test indicate the statistical significance of the experiment. We mark one star (*) if the p-value is lower or close to 0.05. Best results are highlighted in **bold**.

(a) Percentage of good

# Atoms	P1	P2	P3	P4	P5	P6	Avg.	Agr.	P-value
10	10	12	14	8	26	24	15.7	94.5	5.86 E-02*
100	34	40	46	34	64	58	46.0	94.5	5.86 E-02*
1000	84	82	86	78	**100**	80	85.0	94.1	1.58 E-01
5000	**100**	**98**	**98**	**90**	**100**	86	**95.3**	91.7	**-**
10000	96	96	94	88	**100**	**90**	94.0	91.7	2.06 E-01

(b) Percentage of best

# Atoms	P1	P2	P3	P4	P5	P6	Avg.	Agr.	P-value
10	0	2	4	0	0	0	1.0	77.1	6.25 E-5**
100	6	6	4	6	4	2	4.7	76.9	8.16 E-5**
1000	30	38	32	30	44	34	34.7	73.5	2.45 E-3**
5000	**54**	**56**	**56**	**52**	**48**	**50**	**52.7**	71.2	**-**
10000	46	48	52	46	**48**	46	47.7	74.3	2.20 E-1

Relation between prediction performance and explainability. Throughout Fig. 7, Table 12, and Table 13, we could not find concrete evidence for a trade-off between explainability and prediction performance. Rather, we found models with good explainability also produce good prediction performance (i.e., models with antecedent length 2 to 4, models having number of atoms $1,000$ or more atoms). This is consistent with our framework $p(y|x,b) = \sum_{\boldsymbol{\alpha}} p(y|\boldsymbol{\alpha})p(\boldsymbol{\alpha}|x,b)$, which passes information from input to prediction only via explanations, as opposed to other unexplainable parts. Thus, the expressivity of explanations and the capacity of the model are tightly related. If the hyperparameter settings significantly constrain the expressivity of the explanations (such as limiting the number of atoms to 10), both explanation quality and predictive performance will decrease significantly.

C.6 Explanation Stability

Do explanations keep the same in different runs? We conduct experiments to confirm that our model usually generates unique explanations for the same instances in different runs. Comparing the model explanations trained with 5 seeds reveals that, on average, 90.04% of atoms were shared by explanations from different seeds, and 71.27% were identical on Yelp. This comparison suggests that our model generates a unique explanation for the same instance, even in the absence of a direct controlling factor. The reason why we can generate unique explanations is that we optimize the explanation generator with two globally consistent rewards in Eq. 2: 1) human's prior belief about which explanation types are good and 2) the explanation (rule) confidence that is measured by the global prediction accuracy over the entire training corpus given the rule. Since the second reward is a real number instead of a discrete value and has a globally consistent meaning, the optimal explanation is usually unique and stable, leading to similar results when trained with different random seeds.

Do similar instances lead to similar explanations? Table 14 shows examples of generated explanations for similar inputs. SELOR successfully maintains its explanation when minor changes are made to input words, but suggests a new explanation when critical changes are made. In case (a), for example, our method provides the same explanation when the words "*pizza*" and "*waiters*" are changed to "*pasta*" and "*servers*". However, when sentiment-related words such as "*cold*" and "*rude*" are changed, it adapts to the new words and gives a new explanation.

Table 14: Generated explanations of samples and their perturbation. The manually changed words are highlighted in **bold**

Case	Sample	Model Explanation	Prediction
(a)	This place is awful. The pizza was cold, and the waiters were rude.	awful, cold, rude	Negative
	This place is awful. The **pasta** was cold, and the **servers** were rude.	awful, cold, rude	Negative
	This place is awful. The pizza was **undercooked**, and the waiters were **unfriendly**.	awful, undercooked, unfriendly	Negative
(b)	I love here. It was an amazing experience to eat a cheesy macaroni.	love, amazing, cheesy	Positive
	I **recommend** here. It was **a happy** experience to eat a cheesy macaroni.	recommend, happy, cheesy	Positive
	I **hate** here. It was **a bad** experience to eat a cheesy macaroni.	hate, bad, experience	Negative
	I ordered three tacos and all 3 were downright lousy. Can't remember the last time I had food this bad. The shrimp taco was overbreaded and in a sickly sweet sauce, the shredded beef taco was very tiny and thankfully, I can't remember what the third taco tasted like. To the reviewer who posted that these tacos are top notch.... what are you smoking? I waited forever to get my food and saw numerous other people who came in after me get their food. Waiter was MIA. Not coming back....ever.	lousy, bad, waited, forever	Negative

(c)	I ordered three tacos and all 3 were downright lousy. Can't remember the last time I had food this bad. The shrimp taco was overbreaded and in a sickly sweet sauce, the shredded beef taco was very tiny and thankfully, I can't remember what the third taco tasted like. To the reviewer who posted that these tacos are top notch.... what are you smoking? I waited **a little** to get my food and saw numerous other people who came in after me get their food. Waiter was MIA. Not coming back....ever.	lousy, bad, waited, not	Negative
	I ordered three **awful, terrible** tacos and all 3 were downright lousy. Can't remember the last time I had food this bad. The shrimp taco was overbreaded and in a sickly sweet sauce, the shredded beef taco was very tiny and thankfully, I can't remember what the third taco tasted like. To the reviewer who posted that these tacos are top notch.... what are you smoking? I waited forever to get my food and saw numerous other people who came in after me get their food. Waiter was MIA. Not coming back....ever.	awful, terrible, waited, forever	Negative
(d)	I had an amazing 4 course meal here with my family from philadlephia. my father runs a farmers market there and was very impressed with their use of seasonal and local foods. We had an amazing pork belly salad and I had duck wrapped in bacon and stuffed with pate, which sounds insanely heavy, but it was not; the portion was small enough not to be overwhelmed and it was not overly greasy at all. It was a fantastic meal. I think l'etoile is on par with top restaurants in bigger cities.	amazing, family, stuffed, fantastic	Positive
	I had **a great** 4 course meal here with my family from philadlephia. my father runs a farmers market there and was very impressed with their use of seasonal and local foods. We had an amazing pork belly salad and I had duck wrapped in bacon and stuffed with pate, which sounds insanely heavy, but it was not; the portion was small enough not to be overwhelmed and it was not overly greasy at all. It was a fantastic meal. I think l'etoile is on par with top restaurants in bigger cities.	great, family, amazing, fantastic	Positive
	I had an **awful** 4 course meal here with my family from philadlephia. my father runs a farmers market there and was very **disappointed** with their use of seasonal and local foods. We had a **terrible** pork belly salad and I had duck wrapped in bacon and stuffed with pate, which sounds insanely heavy; **and it was right**; the portion was **too small to be full** and it **was overly** greasy at all. It was a **bad** meal. I think l'etoile is on par with **bad** restaurants in bigger cities.	awful, disappointed, terrible, bad	Negative