

Interpreting Deep Learning Models in Natural Language Processing: A Review

Xiaofei Sun¹, Diyi Yang², Xiaoya Li¹, Tianwei Zhang³,
Yuxian Meng¹, Han Qiu⁴, Guoyin Wang⁵, Eduard Hovy⁶, Jiwei Li^{1,7}

¹Shannon.AI, ²Georgia Institute of Technology

³Nanyang Technological University, ⁴Tsinghua University

⁵Amazon Alexa AI, ⁶Carnegie Mellon University, ⁷Zhejiang University

Abstract

Neural network models have achieved state-of-the-art performances in a wide range of natural language processing (NLP) tasks. However, a long-standing criticism against neural network models is the lack of interpretability, which not only reduces the reliability of neural NLP systems but also limits the scope of their applications in areas where interpretability is essential (e.g., health care applications). In response, the increasing interest in interpreting neural NLP models has spurred a diverse array of interpretation methods over recent years. In this survey, we provide a comprehensive review of various interpretation methods for neural models in NLP. We first stretch out a high-level taxonomy for interpretation methods in NLP, i.e., training-based approaches, test-based approaches and hybrid approaches. Next, we describe sub-categories in each category in detail, e.g., influence-function based methods, KNN-based methods, attention-based models, saliency-based methods, perturbation-based methods, etc. We point out deficiencies of current methods and suggest some avenues for future research. ¹

1 Introduction

Deep learning based models have achieved state-of-the-art results on a variety of natural language processing (NLP) tasks such as tagging [127, 165, 217, 120], text classification [100, 221, 144, 121, 32, 199], machine translation [194, 59, 222, 187], natural language understanding [210, 110, 181, 200], dialog [4, 207, 16], and question answering [93, 97, 218, 208]. By first transforming each word (or each *token* in the general sense) into its word vector and then mapping these vectors into higher-level representations through layer-wise interactions such as recursive nets [175, 82, 117, 27], LSTMs [77], CNNs [95, 100], Transformers [194], deep neural networks are able to encode rich linguistic and semantic knowledge in the latent vector space [130, 41, 78, 71], capture contextual patterns and make accurate predictions on downstream tasks. However, a long-standing criticism against deep learning models is the lack of interpretability: in NLP, it is unclear how neural models deal with *composition* in language, such as implementing affirmation, negation, disambiguation and semantic combination from different constituents of the sentence [86, 203, 188]. The *uninterpretability* of deep neural NLP models limits their scope of applications which require strong controllability and interpretability.

Designing methods to interpret neural networks has gained increasing attentions. in the field of computer vision, interpreting neural models involves identifying the part of the input that contributes most to the model predictions, and this paradigm has been widely studied (CV), using saliency maps [172, 211], layer-wise relevance propagation [13, 132, 102], model-agnostic interpreting methods [152] and back-propagation [171]. These methods have improved neural model interpretability and reliability from various perspectives, and raised the community’s awareness of the necessity of neural model interpretability when we want to build trustworthy and controllable systems [2, 191, 131]. Different from computer vision, the basic input units in NLP for neural models are discrete language tokens rather than continuous pixels in images [20, 38, 15]. This discrete nature of language poses a challenge for interpreting neural NLP models, making the interpreting methods in CV hard to be directly applied to NLP domain [22, 114]. To accommodate the discrete nature of texts, a great variety of works have rapidly emerged over the past a few years for neural model interpretability. A systematic review is thus needed to sort out and compare different

¹E-mail addresses: xiaofei_sun@shannonai.com (X. Sun), diyi.yang@cc.gatech.edu (D. Yang), xiaoya_li@shannonai.com (X. Li), tianwei.zhang@ntu.edu.sg (T. Zhang), yuxian_meng@shannonai.com (Y. Meng), qiuhan@tsinghua.edu.cn (H. Qiu), guoyiwan@amazon.com (G. Wang), hovy@cmu.edu (E. Hovy), jiwei_li@shannonai.com (J. Li).

Training-based methods	Identify training instances responsible for the prediction of the current test instance.
Test-based methods	Identify which part(s) of a test instance responsible for the model prediction.
Hybrid methods	Identify because of which training instances, the model attends to which part(s) of the test instance for its prediction.

Table 1: Summary of the high-level categories of interpretation methods for neural NLP models.

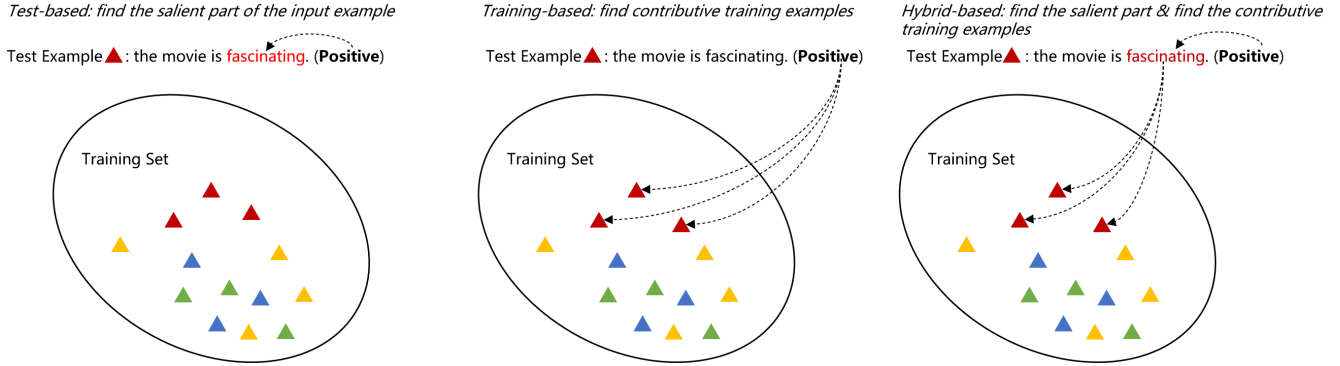


Figure 1: Difference of training-based, test-based and hybrid interpreting methods. The figure is adapted from the Figure 1 in the work of [129]. In a nutshell, test-based methods target finding the most salient part(s) of the input; training-based methods find the most contributive training examples responsible for a model’s predictions; hybrid methods combine both worlds, by first finding the most salient part(s) of the input and then identifying the training instances responsible for the salient part(s).

methods, providing a comprehensive understanding of from which perspective a neural NLP model can be interpreted and how the model can be interpreted effectively.

In this survey, we aim to provide a comprehensive review of existing interpreting methods for neural NLP models. We propose taxonomies to categorize existing interpretation methods from the perspective of (1) *training-focused* v.s. *test-focused*; and (2) *joint* v.s. *post-hoc*. Regarding training-focused v.s. test-focused, the former denotes identifying the training instances or elements within training instances responsible for the model’s prediction on a specific test example, while the latter denotes finding the most salient part of the input test-example responsible for a prediction. The two perspectives can be jointly considered: aiming to identify because of which training instances, the model attends to which part(s) of the test example for its prediction. Based on the specific types of algorithms used, training-based methods can be further divided into influence-function based methods, KNN-based methods, kernel-based methods, etc. Test-based methods can be divided into saliency-based methods, perturbation-based methods, attention-based methods, etc. Figure 1 depicts the difference of training-based, test-based and hybrid interpreting methods. Regarding joint v.s. post-hoc, **for the former joint approach, a joint interpreting model is trained along with the main classification model and the interpreting tool is nested within the main model, such as attention; for the latter post-hoc approach, an additional probing model, which is separately developed after the main model completes training, is used for interpretation.** Figure 2 depicts the difference between joint and post-hoc approaches. Categorization with taxonomy framework naturally separates different methods and makes audience easy to locate the works of interest. Table 1 shows the high-level categories of explanations

We analyze the literature with respect to these dimensions and review some of the representative techniques regarding each of these categories. In summary, this work:

- Categorizes interpreting methods according to *training-focused* v.s. *test-focused* and *joint* v.s. *post-hoc*.
- Provides an empirical review of up-to-date methods for interpreting neural NLP models.
- Discusses deficiencies of current methods and suggest avenues for future research.

The rest of this survey is organized as follows: Section 2 clarifies the high-level taxonomy proposed in this work. Section 3, 4 and 5 review existing training-based methods, test-based methods and hybrid methods for interpreting neural NLP models. Section 6 discusses some open problems and directions that may be intriguing for future works. Section 7 summarizes this survey.

Joint methods jointly train the main classification model and the interpretation component.

Pos-hoc methods first train the main model, and then apply a separate interpretation model to the trained main model for interpretation.

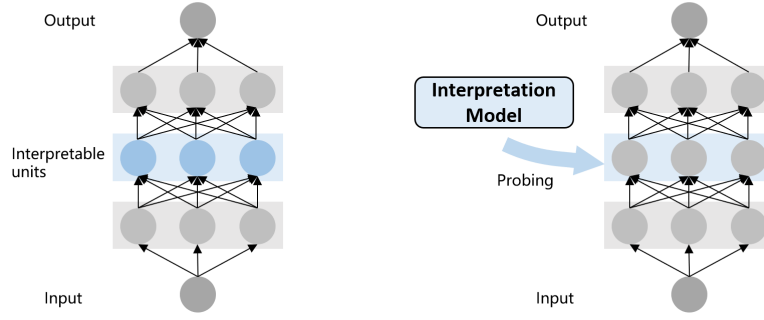


Figure 2: An illustrative comparison between *joint* interpreting methods and *post-hoc* interpreting methods.

1.1 Related Surveys

Related to this work, [216] conducted a survey of neural network interpretability methods used in computer vision, bioinformatics and NLP along several different dimensions such as using passive/active interpretation approaches or focusing on local/global effect. However, our survey provides an in-depth overview for interpreting neural NLP models for NLP tasks. Another line of work, AllenNLP Interpret [198], provides interpretation primitives (e.g., input gradients) for AllenNLP model and task, a suite of built-in interpretation methods, and a library of front-end visualization components; the toolkit mainly supports gradient-based saliency maps and adversarial attacks. In contrast, our review covers more comprehensive interpretability methods.

2 High-level Taxonomy

In this section, we provide a high-level overview of the taxonomy used in this survey. As shown in Table 2, the topmost level categorizes existing methods by whether they aim to interpret models from the training side, from the test side, or from both sides.

Walking into the second level, this taxonomy categorizes literature based on the basic tools they use to interpret model behaviors. For training-based methods, they are sub-categorized into influence functions (Section 3.1), KNNs for interpretation (Section 3.2) and kernel-based explanation (Section 3.3), which are respectively related to using influence functions, k nearest neighbors and kernel-based models for interpretation in the training side. Test-based methods (Section 4) are sub-categorized into saliency maps (Section 4.1), attention as explanation (Section 4.2.1), and explanation generation (Section 4.3), which respectively use saliency maps, attention distributions and explainable contexts to interpret models in the test side. Hybrid methods interpret model predictions by jointly examining training history and test stimuli, which can be viewed as a combination of training-based and test-based methods (Section 5). Each category is paired with whether they belong to *joint* or *post-hoc*. In the following section of this survey, we will expound on some of the representative works from each of the categories.

3 Training-Based Interpretation

The main purpose of training-based methods is to identify **which training instances or units of training instances in the training set are responsible for the model's prediction at a specific test example**. Existing training-based techniques fall into the following three categories:

- **Influence function based interpretation:** using influence functions to estimate the *influence* of each training point on an input test example. The training points with the highest influences are viewed as the most contributive instances.
- **Nearest Neighbor based interpretation:** extracting the k nearest neighbor points in the training set that are close to an input test example. Extracted nearest neighbors are viewed as the most contributive instances
- **Landmark based interpretation:** An input test sentence is fed to the classifier along with *landmarks*, which refer to a set of training examples used to compile the representation of any unseen test instance. The influence of training examples on an input test example is measured by the contribution of each landmark to the model's decision on the

First Level	Second Level	Technique Description	Joint/Post-hoc	Representative Works
Training-based	Influence functions	Measuring the change in model parameters when a training point changes.	Post-hoc	[70, 103, 209, 69]
	KNNs for interpretation	Searching k nearest neighbors over a model’s hidden representations to identify training examples closest to a given evaluation example.	Post-hoc	[197, 148]
	Kernel-based explanation	Within the kernel-based framework, selecting landmarks that the Layer-wise Relevance Propagation algorithm highlights as the most active elements responsible for the model prediction.	Joint/Post-hoc	[47, 48]
Test-based	Saliency maps	Computing the saliency score of each token or span based on network gradient or feature attribution. The part of sentence with the highest salience score is viewed as the interpretable content.	Post-hoc	[116, 118, 184, 156, 6, 140, 61, 68, 99, 35]
	Attention as explanation	Leveraging attention distributions over tokens as a tool of interpretation. The most attentive token or span is regarded as the most contributive to the model prediction.	Joint	[182, 86, 164, 203, 193, 154, 63, 41, 195, 190, 78, 30, 143, 71, 150]
	Explanation generation	Generating or inducing explanations for the model’s prediction. The explanation may come from a subset of the input sentence, from external knowledge or from a language model that generates explanations from scratch.	Joint	[113, 134, 33, 185, 53, 87, 107, 149, 124, 56, 40, 79, 19, 161, 50, 153, 147]
Hybrid		Interpreting model behaviors by jointly examining training history and test stimuli.	Post-hoc	[129]

Table 2: An overview of high-level taxonomy for interpretation methods in NLP. Column 1 denotes the topmost level of the taxonomy. Column 2 denotes the second-level categorizations. Column 3 gives brief descriptions of the techniques grouped in the second-level categorizations. Column 4 notifies whether the body of work is *joint* (the interpreting model is trained jointly with the main model) or *post-hoc* (the interpreting method is applied to the trained main model). Column 5 contains a list of representative references.

test example based on the Layer-wise Relevance Propagation (LRP) [14] algorithm. As landmarks are organized via kernel-based deep architectures, this method is also referred to as *Kernel-based Explanation*.

We will describe these methods at length in the rest of this subsection.

3.1 Influence Functions Based Interpretation

3.1.1 Influence functions

The *influence* of a training point can be defined as the change in the model parameters when a specific training point is removed. To measure the change of model parameters, an intuitive way is to retrain the model on the training set with the training example left-out. However, this is prohibitively time-intensive as we need to retrain the model from scratch for every training example. Alternatively, *influence functions* [45, 44, 104] provide an approximate but tractable way to track parameter change without the need to repeatedly retrain the model.

Concisely speaking, the influence function estimates the impact on the model parameters when the loss of a particular training point is up-weighted by a small perturbation ϵ . This can be done through first-order approximation. Let $z_i (i = 1, 2, \dots, n)$ be a training point in the training set. The model parameters θ is learned to minimize a given loss function L :

$$\hat{\theta} = \arg \min_{\theta} \frac{1}{n} \sum_{i=1}^n L(z_i; \theta) \quad (1)$$

Analogously, the optimal model parameters $\theta_{z,\epsilon}$ when the loss of a training point z is weighted by a small perturbation can be

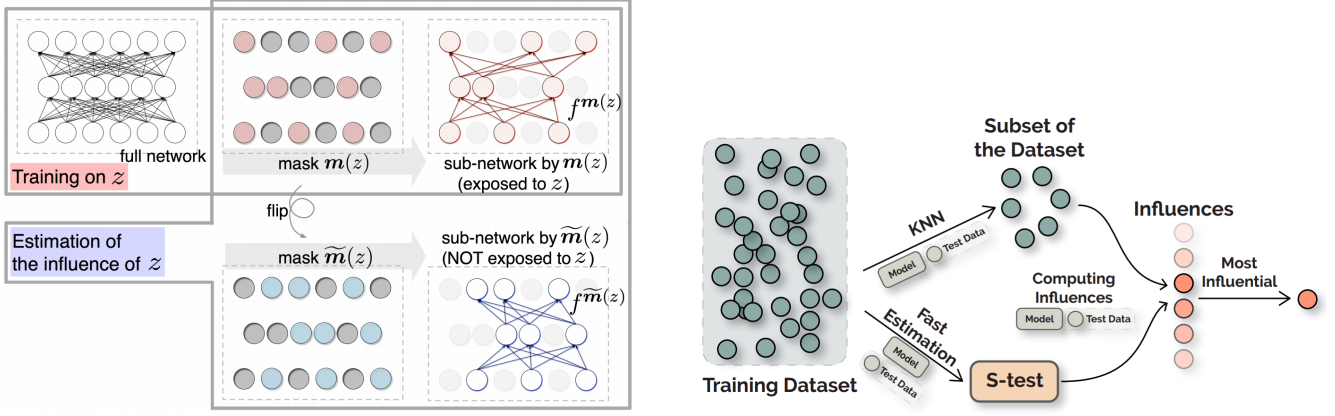


Figure 3: Efficient implementations of influence functions proposed in [103] (left) and [69] (right). The figures are borrowed from the original literature.

derived by minimizing the following loss:

$$\hat{\theta}_{z,\epsilon} = \arg \min_{\theta} \frac{1}{n} \sum_{i=1}^n L(z_i; \theta) + \epsilon L(z; \theta) \quad (2)$$

By referring the audience to [104] for detailed proofs, the influence of a training point z on the model parameters θ , which is denoted by $I(z, \theta)$, is given as follows:

$$I(z, \theta) \triangleq \frac{d\hat{\theta}_{z,\epsilon}}{d\epsilon} = -H_{\theta}^{-1} \nabla_{\theta} L(z; \hat{\theta}) \quad (3)$$

where $H_{\theta} \triangleq \frac{1}{n} \sum_{z_i} \nabla_{\theta}^2 L(z_i; \hat{\theta})$ is the Hessian matrix over all training instances z_i in the training set, and $L(z_i; \hat{\theta})$ is the loss of z over model parameters θ . Equation 3 expresses the degree to which a small perturbation on a training point z influences the model parameters θ . Based on Equation 3, [70] applied the chain rule to measure how this change in model parameters would in turn affect the loss of the test input x :

$$I(z, x) \triangleq \frac{dL(x; \hat{\theta})}{d\epsilon_z} = \nabla_{\theta} L(x; \hat{\theta}) \cdot \frac{d\hat{\theta}_{z,\epsilon}}{d\epsilon} = \nabla_{\theta} L(x; \hat{\theta}) \cdot I(z, \theta) = -\nabla_{\theta} L(x; \hat{\theta})^{\top} H_{\theta}^{-1} \nabla_{\theta} L(z; \hat{\theta}) \quad (4)$$

[70] defined the negative score $-I(z, x)$ as the *influence score*. Equation 4 reveals that a positive influence score for a training point z with respect to the input example x means that the removal of z from the training set would be expected to cause a drop in the model’s confidence when making the prediction on x . By contrast, a negative influence score means that the removal of z from the training set would be expected to give an increase in the model’s confidence when making this prediction. Experiments and analysis from [70] validated the reliability of influence functions when applied to deep transformer-based [194] models.

3.1.2 Efficient influence functions

A closer look at Equation 4 reviews the disadvantage of influence functions: the computational cost of computing the inverse-Hessian matrix H_{θ}^{-1} and iterating over all training points z to find the most (least) influential example is infeasible to bear for large-scale models such as BERT [52]. Efficient estimates for influence functions are thus needed to improve runtime efficiency while preserving the interpretability. [103] and [69] respectively address this issue from the model side and from the algorithm side. Figure 3 illustrates how these two methods work. [103] proposed *turn-over dropout*, a variant of dropout applied to the network for each individual training instance. After training the model on the entire training dataset, *turn-over dropout* generates an individual-specific dropout mask scheme $m(z)$ for each training point z , and then feeds z into the model for training using the proposed mask scheme $m(z)$. The mask scheme $m(z)$ corresponds to a sub-network $f^m(z)$, which is updated when the model is trained on z , but the counter-part of the model $f^{\tilde{m}(z)}$, is not at all affected. Therefore, the two sub-networks, $f^m(z)$ and $f^{\tilde{m}(z)}$, can be analogously perceived as two different networks trained on a dataset with or without z . The influence of z on a test input example x can thus be computed as follows:

$$I(z, x) \triangleq L(x; f^m(z)) - L(x; f^{\tilde{m}(z)}) \quad (5)$$

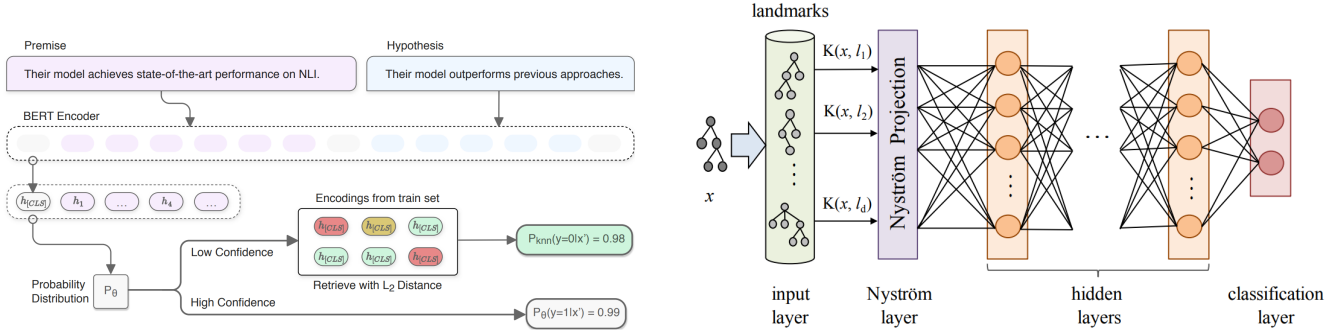


Figure 4: Overview of KNN-based (left, the figure is borrowed from the work of [148]) and kernel-based (right, the figure is borrowed from the work of [48]) interpretation methods. KNN-based methods first cache high-dimensional representation of training instances and search for the k nearest neighbors with respect to the test example in the latent space. Kernel-based methods leverage *landmark* instances from the training set as part of the model input, providing linguistic evidence for interpretation.

Experimenting on the Stanford Sentiment TreeBank (SST-2) [176] and the Yahoo Answers dataset [214] with the BERT model, the *turn-over dropout* technique is able to interpret erroneously classified test examples by identifying the most influential training instances, with a more efficient computation process. [69] tackled the computation issue of the original influence functions from the algorithm perspective. Their optimization process can be divided into three aspects: (1) k NN constraint on the search space; (2) the inverse Hessian computation speedup; and (3) parallelization. First, to avoid the global search on the entire training dataset for the most influential data point, the search space is constrained within a small subset of the training set. This can be achieved by selecting the top- k nearest neighbors to the test input instance x based on the l_2 distance between extracted features. Second, to speed up computation of the inverse Hessian matrix H_θ^{-1} , [69] carefully selected the required hyperparameters used to compute the inverse Hessian matrix so that the inverse Hessian-vector product $H_\theta^{-1} \nabla_\theta L(x; \hat{\theta})$ (Equation 4) can efficiently be calculated. Third, [69] applied off-the-shelf multi-GPU parallelization tools to compute influence scores in a parallel manner. The confluence of all three factors leads to a speedup rate of 80x while being highly correlated with the original influence functions. With the new fast version of influence functions, a few applications that are concerned with model interpretation but were previously intractable are demonstrated, including examining the “explainability” of influential data-points, visualizing data influence-interactions, correcting model predictions using original training data, and correcting model predictions using data from a new dataset. The efficient implementation of influence functions provides possibility for new interpretable explorations that scale with the data size and model size.

The influence function based interpreting methods are along the *post-hoc* line, which means that they are applied for interpretation after the main model is fully trained.

3.2 KNNs Based Interpretation

KNN-based interpretation [197, 148] retrieves k nearest training instances that are the closest to a specific test input from the training set. After the training process completes, each training point is passed to the trained model to derive its high-dimensional representation z , which is cached for test-time retrieval. Then at test time, the model outputs the representation of the input example x , uses it as the query to search for k nearest training instances in the cache store. In the meantime, the distance between the query and each retrieved training point can be calculated, and the corresponding ground-truth labels for the retrieved training points can also be extracted. The normalized distances are treated as the weight associated with each training point the model retrieves to interpret its prediction for the test input example, and the percentage of nearest neighbors belonging to the predicted class, which is called the *conformity score* [197], can be interpreted as how much the training data supports a classification decision. [148] uses the conformity score to calibrate an uncertain model prediction, showing that KNN interpretation is able to identify mislabeled examples. Figure 4 (left) provides an overview of the KNN-based interpretation method proposed in [148].

KNN based interpretation works after the training process completes, and thus belongs to the *post-hoc* category.

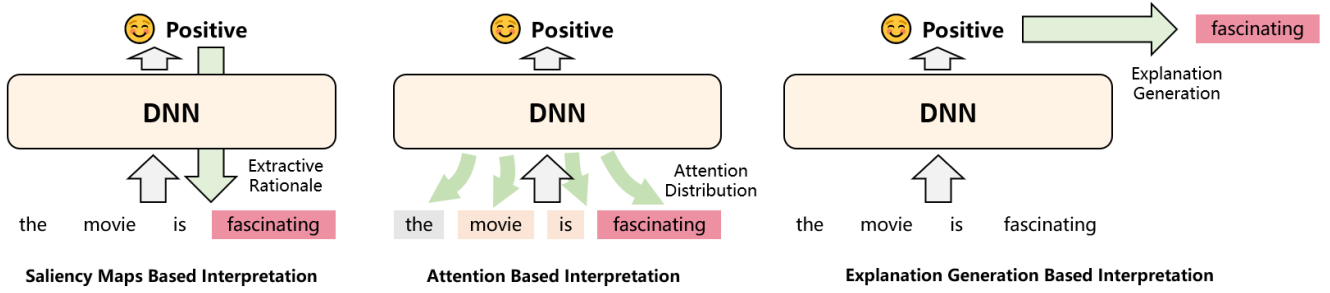


Figure 5: An overview of test-based methods. *Left:* The saliency score is the importance or relevance of a token or a span with respect to the model prediction. A higher saliency score denotes greater importance. *Middle:* Attention scores over input tokens reflects how the model distributes its attention to different parts of the input. Attention distributions can thus be viewed as a tool for interpretation. *Right:* Given the input, the model gives its prediction as well as the evidence supporting its decision. A judicious decision can be explained in a way of generating the corresponding explanations.

3.3 Landmark Based Interpretation

Landmark based interpreting methods [47, 48] use *landmarks*, which are a set of real reference training examples, to compile the linguistic properties of an unseen test example. This line of methods combine the Layerwise Relevance Propagation (LRP) [13] and Kernel-based Deep Architectures (KDAs) [46] to complete interpretation. An illustration is shown in Figure 4 (right).

More formally, given an input example x and d landmarks $\{l_1, l_2, \dots, l_d\}$ sampled from the training set, the model first computes the similarity score $K(x, l_i)$ between x and each landmark l_i using a kernel function $K(\cdot, \cdot)$. Then, a Nyström layer [204] is used to map the vector of similarity scores $[K(x, l_1), K(x, l_2), \dots, K(x, l_d)]$ to a high-dimensional representation. Passing through multiple ordinary hidden layers, the model last makes its prediction at the classification layer. The expected explanation is obtained from the network output by applying LRP to revert the propagation process, therefore linking the output back to the similarity vector. Once LRP assigns a score to each of these landmarks (i.e., the corresponding similarity score $K(x, l_i)$), we can select the *positively active* landmarks as evidence in favor of a class C . A template-based natural language explanation is generated based on the activated landmarks. As an example given in [48], the generated explanation to the input example “What is the capital of Zimbabwe?” that refers to *Location* is “since it recalls me of ‘What is the capital of California?’, which also refers to *Location*”. In this example, “What is the capital of California?” is the activated landmark used to trigger the template-based explanation, and *Location* is the class C the model predicts.

To decide whether the landmark-based model belonging to the joint or post-hoc category, we need to take into consideration its two constituent components: (1) the kernel architecture component which involves the kernel function $K(\cdot, \cdot)$ and the Nyström layer, and (2) the LRP scoring mechanism. For the former, the kernel architecture is trained together with the entire model and therefore the landmark-based model can be viewed as along the *joint* interpretation line. For the latter, when computing the importance of each landmark, LRP is a typical *post-hoc* interpreting method. The landmark-based model can thus be viewed as both joint and post-hoc.

4 Test-Based Interpretation

Different from training-based methods that focus on interpreting neural NLP models by identifying the training points responsible for the model’s prediction on a specific test input, the test-based methods, instead, **aim at providing explanations about which part(s) of the test example contribute most to the model prediction**. This line of methods are rooted in the intuition that not every token or span within the input sentence contributes equally to the prediction decision, and thus the most contributive token or span can usually interpret the model behaviors. For example in Figure 1, the input test example “the movie is fascinating” has a very typical adjective “fascinating” that would nudge the model to predict the class label *Positive*. In this sense, the most contributive part is word “fascinating”. If we insert the negation “not” before “fascinating”, the most contributive part would therefore be “not fascinating”, a span rather than a word in the input sentence.

Most of existing test-based methods fall into the following three categories: saliency maps, attention as explanation and explanation generation. Saliency maps and attention provide very straightforward explanations in the form of visualized heatmaps distributed to the input text, showing intuitive understandings of which part(s) of the input the model focuses most on, and which part(s) of the input the model pays less attention to. The methods of textual explanation generation justify the model prediction by generating causal evidence either from a part of the input sentence, from external knowledge resources or

Method	Formulation
Vanilla Gradient	$\text{Score}(x_i) = \frac{\partial S_y(x)}{\partial x_i}$
Integrated Gradient	$\text{Score}(x_i) = (x_i - \bar{x}_i) \cdot \int_{\alpha=0}^1 \frac{\partial S_y(\tilde{x})}{\partial \tilde{x}_i} \big _{\tilde{x}=\bar{x}+\alpha(x-\bar{x})} d\alpha$
Perturbation	$\text{Score}(x_i) = S_y(x) - S_y(x \setminus \{x_i\})$
LRP	$\text{Score}(x_i) = r_i^{(0)}, r_i^{(l)} = \sum_j \frac{z_{ji}}{\sum_{i'} (z_{ji'} + b_j) + \epsilon \cdot \text{sign}(\sum_{i'} (z_{ji'} + b_j))} r_j^{(l+1)}$
DeepLIFT	$\text{Score}(x_i) = r_i^{(0)}, r_i^{(l)} = \sum_j \frac{z_{ji} - \bar{z}_{ji}}{\sum_{i'} z_{ji'} - \sum_{i'} \bar{z}_{ji'}} r_j^{(l+1)}$

Table 3: Mathematical formulation of different saliency methods. LRP and DeepLIFT perform in a top-down recursive manner.

completely from scratch. This test-based framework of model interpretation, in contrast to training-based methods, is concerned with one particular input example, and conveys useful and intuitive information to end-users even without any AI background.

Figure 5 shows a high-level overview for the three different strands of test-based methods. In the rest of this subsection, we will describe each line of methods in detail.

4.1 Saliency-based Interpretation

In natural languages, some words are more *important* than other words in indicating the direction to which the model predicts. In sentiment classification for example, the words associated with strong emotions such as “*fantastic*”, “*scary*” and “*exciting*” would provide interpretable cues for the model prediction. If we could measure the *importance* of a word, or a span, and examine the relationship of the most important part(s) with the model prediction, we will be able to interpret the model behaviors through the lens of these important contents and their relationship with model predictions. **The importance of each token (or its corresponding input feature), which sometimes is also called the *saliency* or the *attribution*, defines the relevance of that token with the model prediction.** By visualizing the *saliency maps*, i.e., plotting the saliency scores in the form of heatmaps, users can easily understand why model make decisions by examining the most salient part(s).

We describe four typical ways of computing the saliency score in the rest of this subsection. These methods include: gradient-based, perturbation-based, LRP-based and DeepLIFT-based. Table 3 summarizes the mathematical formulations of these methods.

4.1.1 Gradient-based saliency scores

Gradient-based saliency methods compute the importance of a specific input feature (e.g., vector dimension, word or span) based on the first-order derivative with respect to that feature. Suppose that $S_y(x_i)$ is output with respect to class label y before applying the softmax function, and x_i is the input feature, which is the i th dimension of the word embedding $x = \{x_1, x_2, \dots, x_i, \dots, x_N\}$ for word w in NLP tasks. We have:

$$S_y(x_i + \Delta x_i) \approx S_y(x_i) + \frac{\partial S_y(x_i)}{\partial x_i} \Delta x_i \quad (6)$$

$\frac{\partial S_y(x_i)}{\partial x_i}$ can be viewed as the change of $S_y(x_i)$ with respect to the change of x_i . If $\frac{\partial S_y(x_i)}{\partial x_i}$ approaches 0, it means that $S_y(x_i)$ is not at all sensitive to the change of x_i . $\frac{\partial S_y(x_i)}{\partial x_i}$ can thus be straightforwardly viewed as the importance of the word dimension x_i [116]:

$$\text{Score}(x_i) = \frac{\partial S_y(x)}{\partial x_i} \quad (7)$$

Equation 7 computes the importance of a single dimension for the word vector. To measure the importance of a word $\text{Score}(w)$, we can use the norm as follows:

$$\text{Score}(w) = \sqrt{\sum_i \left\| \frac{\partial S_y(x_i)}{\partial x_i} \right\|^2} \quad (8)$$

or multiply the important vector with the input feature vector [51, 6]:

$$\text{Score}(w) = \left| \sum_i x_i \frac{\partial S_y(x_i)}{\partial x_i} \right| \quad (9)$$

SQUAD

Context: QuickBooks sponsored a “Small Business Big Game” contest, in which Death Wish Coffee had a 30-second commercial aired free of charge courtesy of QuickBooks. **Death Wish Coffee** beat out nine other contenders from across the United States for the free advertisement.

Question:

What company won free advertisement due to QuickBooks contest ?
 What company won free advertisement due to QuickBooks ?
 What company won free advertisement due to ?
 What company won free due to ?
 What won free due to ?
 What won due to ?
 What won due to
 What won due
 What won
 What

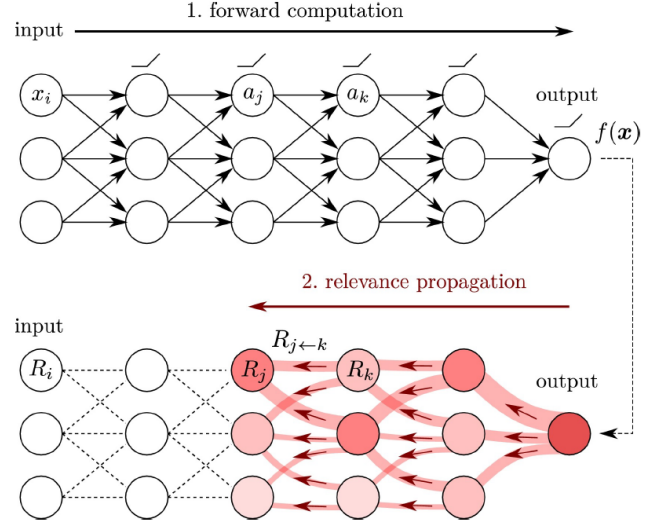


Figure 6: Diagram of the perturbation based method (left, the figure is brought from [61]) and LRP (right, the figure is brought from [64]). *Left*: At each step, the least important word is removed by the saliency score using the leave-one-out technique. As shown in the figure, the heatmaps generated with leave-one-out shift drastically: the word “advertisement” is the most salient in step one and two, but becomes the least salient in step three. *Right*: LRP provides decomposition of the attribute scores from the model prediction to the input features. It does not utilize partial derivatives but rather directly decomposes the activation function to produce the attribute score.

Both methods work well for the interpretation purpose. Albeit simple and easy to implement, this kind of vanilla gradient saliency score assumes a linear approximation of the relationship between the input feature and the model output, which is not the case for deep neural networks. [184] proposed the *integrated gradient (IG)*, a modification to the vanilla gradient saliency score. IG computes the average gradient along the linear path of varying the input from a baseline value \bar{x} to itself x , which produces a more reliable and stable result compared to the vanilla gradient approach. The baseline value is often set to zero. IG can be formulated as the following equation:

$$\text{Score}(x_i) = (x_i - \bar{x}_i) \cdot \int_{\alpha=0}^1 \frac{\partial S_y(\tilde{x})}{\partial \tilde{x}_i} \bigg|_{\tilde{x}=\bar{x}+\alpha(x-\bar{x})} d\alpha \quad (10)$$

4.1.2 Perturbation-based saliency scores

Perturbation-based methods compute the saliency score of an input feature by removing, masking or altering that feature, passing the altered input again into the model and measuring the output change. This technique is straightforward to understand: **if a particular input token is important, then removing it will be more likely to causing drastic prediction change and flip the model prediction.**

One simple way to perturb an input is to erase a word from the input, and examine the change in the model’s prediction on the target class label [118]. This technique is called the *leave-one-out* perturbation method. By iterating over all words, we can find the most salient word that leads to the largest prediction change. The saliency score using leave-one-out can be given as follows:

$$\text{Score}(x_i) = S_y(x) - S_y(x \setminus \{x_i\}) \quad (11)$$

The token-level removal can be extended to the span level or the phrase level [155, 206]. Based on the leave-one-out technique, [61] uses input reduction, which iteratively removes the least salient word at a time, to induce the *pathological behaviors* of neural NLP models. An example is shown in Figure 6 (left).

Another way to perturb the input is to inject learnable *interpretable adversaries* into an input feature [160, 68]. By conforming to specific restrictions (e.g., the direction, the magnitude or the distribution) and learning to optimize predefined training objectives, the adversarial perturbations are able to reveal some interpretable patterns within neural NLP models, such as the word relationships, the information of each word discarded when words pass through layers, and how the model evolves during training.

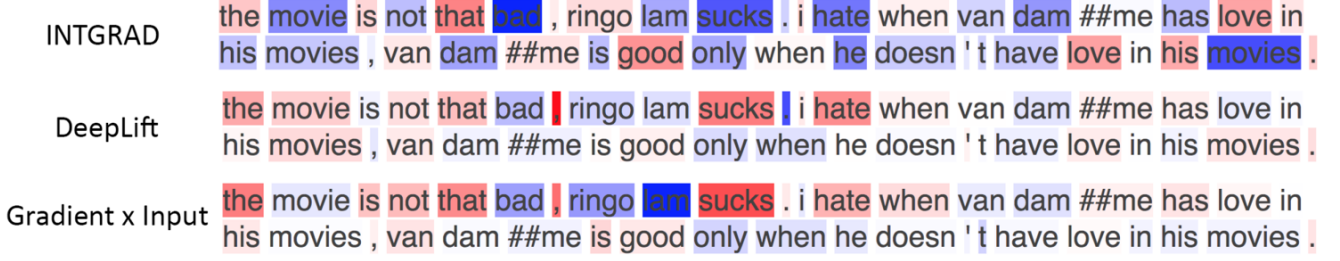


Figure 7: A visualization of saliency methods Integrated Gradient, DeepLIFT and Gradient×Input (the figure is borrowed from [22]). Red denotes a positive saliency value while blue denotes negative values, and the greater the absolute value is, the darker the color would be.

4.1.3 LRP-based saliency scores

Gradient-based and perturbation-based saliency methods directly compute influence of input layer features on the output in an end-to-end fashion, and thus neglects the dynamic changes of word saliency in intermediate layers. Layerwise Relevance Propagation [14], or LRP for short, provides full, layer-by-layer decomposition for the attribute values from the model prediction backward to the input features in a recursive fashion. A sketch of the propagation flow of LRP is shown in Figure 6 (right).

LRP has been applied to interpret neural NLP models for various downstream tasks [9, 10, 11, 54]. LRP defines a quantify $r_i^{(l)}$ – the “relevance” of neuron i in layer l , and starts relevance decomposition from the output layer L to the input layer 0, redistributing each relevance $r_i^{(l)}$ down to all the neurons $r_j^{(l-1)}$ at its successor layer $l - 1$. The relevance redistribution across layers goes as follows:

$$r_i^{(L)} = \begin{cases} S_y(x), & i \text{ is the target unit of the gold class label} \\ 0, & \text{otherwise} \end{cases} \quad (12)$$

$$r_i^{(l)} = \sum_j \frac{z_{ji}}{\sum_{i'} (z_{ji'} + b_j) + \epsilon \cdot \text{sign}(\sum_{i'} (z_{ji'} + b_j))} r_j^{(l+1)}$$

where $z_{ji} = w_{ji}^{(l+1,l)} x_i^{(l)}$ is the weighted activation of neuron i in layer l onto neuron j in layer $l + 1$ and b_j is the bias term. A small ϵ is used to circumvent numerical instabilities. The saliency scores at the input layer can thus be defined as $\text{Score}(x_i) = r_i^{(0)}$. Equation 12 reveals that the more relevant an input feature is to the model prediction, the larger relevance (saliency) score it will gain backward from the output layer. A merit LRP offers is that it allows users to understand how the relevance of each input feature flows and changes across layers. This property enables interpretation from the inside of model rather than the superficial operation of associating input features with the output.

4.1.4 DeepLIFT-based saliency scores

Similar to LRP, DeepLIFT [171, 8] proceeds in a progressive backward fashion, but the relevance score is calculated in a relative view: instead of directly propagating and redistributing the relevance score from the upper layer to the lower layer, DeepLIFT assigns each neuron a score that represents the relative effect of the neuron at the original input x relative to some reference input \bar{x} , taking the advantages of both Integrated Gradient and LRP. The mathematical formulation of DeepLIFT can be expressed as follows:

$$r_i^{(L)} = \begin{cases} S_y(x) - S_y(\bar{x}), & i \text{ is the target unit of the gold class label} \\ 0, & \text{otherwise} \end{cases} \quad (13)$$

$$r_i^{(l)} = \sum_j \frac{z_{ji} - \bar{z}_{ji}}{\sum_{i'} z_{ji'} - \sum_{i'} \bar{z}_{ji'}} r_j^{(l+1)}$$

Using the baseline input \bar{x} , the reference relevance values \bar{z}_{ji} can be computed by running with \bar{x} a forward pass through the neural model. Ordinarily, the baseline is set to be zero. z_{ji} and \bar{z}_{ji} are computed in the same way but use different inputs: $z_{ji} = w_{ji}^{(l+1,l)} x_i^{(l)}$ and $\bar{z}_{ji} = w_{ji}^{(l+1,l)} \bar{x}_i^{(l)}$.

Figure 7 shows the saliency maps generated by three saliency methods: integrated gradient (IG), DeepLIFT and Gradient×Input, with respect to the input sentence “the movie is not that bad, ringo lam sucks. i hate when van dam ##me has love in his movies,

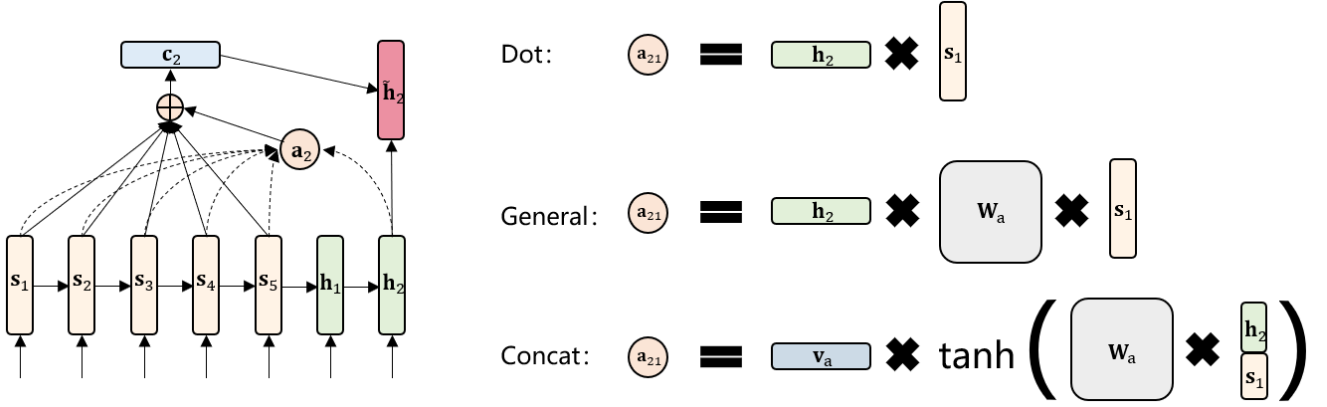


Figure 8: A visualization of the attention mechanism. We show the three forms of attention proposed in [126].

van dam ##me is good only when he doesn ' t have love in this movies.” As can be seen from the figure, the highlighted words are very different across different methods. IG produces meaningful explanations as it correctly marks out the words “bad”, “sucks” and “hate” that strongly guide the model prediction toward negative, whereas DeepLIFT and vanilla gradient struggle to correctly detect the interpretable units. This is a case where IG works well while DeepLIFT and vanilla gradient do not, while in other cases, the superiority of IG is not guaranteed. We thus should take multiple interpreting methods for full consideration.

Because saliency-based interpretation uses an independent probing model to interpret the behaviors of the main model, the methods belong to the *post-hoc* category.

4.2 Attention-based Interpretation

4.2.1 Background

When humans read, they only pay *attention* to the key part. For example, to answer the question “What causes precipitation to fall”, given the document context “In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravity.” humans could easily identify the correct phrase “falls under gravity” in which the answer “gravity” resides, and other parts of the document are less attended.

The attention mechanism, grounded on the way how humans read, has been widely used in NLP [15, 126]. Take the neural machine translation task as an example. Assume the hidden representations of the input document is $s = \{s_1, s_2, \dots, s_n\}$, where each s_i is a d -dimensional vector and n is the input length, and the decoder sequentially generates one token per time step:

$$p(y_j|y_{<j}, s) = \text{softmax}(W\tilde{h}_j), \tilde{h}_j = \text{Attention}(h_j, s), h_j = f(h_{j-1}) \quad (14)$$

W is a transformation matrix that outputs a vocabulary-size vector, f computes the current hidden state given the previous state and can be of any recurrent structure, and Attention is the attention component that takes as input the current hidden state and the source hidden representations and outputs a new attended hidden state, which is then fed to the softmax layer for model prediction. The core idea of attention is to derive a context vector c_j that captures the weighted source hidden representations, and the weight notifies how much should the current time step should pay attention to each of the source input token. This can be formalized as:

$$a_{ji} = \text{align}(h_j, s_i) = \frac{\exp(\text{score}(h_j, s_i))}{\sum_{i'} \exp(\text{score}(h_j, s_{i'}))} \quad (15)$$

Once obtaining the weight a_{ji} , the aggregated hidden state \tilde{h}_j in Equation 14 can be calculate by:

$$\tilde{h}_j = \tanh(W_c[c_j; h_j]), c_j = \sum_{i=1}^n a_{ji}s_i \quad (16)$$

The remaining problem is how to compute the attention weight a_{ji} in Equation 15. [126] proposed three forms of scores: the *dot* form, the *general* form, and the *concat* form. We provide a figure illustration in Figure 8. They can be formalized as follows:

$$\text{score}(h_j, s_i) = \begin{cases} h_j^\top s_i, & \text{dot} \\ h_j^\top W_a s_i, & \text{general} \\ v_a^\top \tanh(W_a [h_j; s_i]), & \text{concat} \end{cases} \quad (17)$$

[194] proposed *self-attention*, where attention is performed between different positions of a *single* sequence rather than two separate sequences in order to compute a representation of the sequence. The expression of self-attention is very like to the dot form of [126], but with differences that each token has three representations – the query q , the key k and the value v , and a scaled factor $\sqrt{d_k}$ is used to avoid overlarge numbers:

$$\text{SelfAttention}(q_j, k_i, v_i) = \frac{\exp(q_j^\top k_i / \sqrt{d_k})}{\sum_{i'} \exp(q_j^\top k_{i'} / \sqrt{d_k})} v_i \quad (18)$$

More importantly, self-attention works in conjunction with the *multi-head* mechanism. Each head represents a sub-network and runs self-attention independently. The produces results are then collected and processed as input to the next layer.

Regardless of different forms of attention, the core idea is the same: *paying attention to the most crucial part in the document*. Given attention weights, we can regard the weight as a measure of importance the model assigns to each part of the input. The following questions arise regarding model interpretation through attention weights: (1) can the learned attention patterns indeed capture *the most crucial* context? (2) How the attention mechanism can be leveraged to interpret model decisions? (3) Does attention reflect some linguistic knowledge when it is applied to neural models? This section describes typical works that target the above three questions.

4.2.2 Is attention interpretable?

There has been a growing debate on the question of *whether attention is interpretable* [164, 143, 86, 203, 193, 30]. The debate centers around whether higher attention weights denote more influence on the the model prediction, which is sometimes unclear.

[164, 143, 86, 30] hold negative attitude towards attention’s interpretability. [86] empirically assessed the degree to which attention weights provide meaningful explanations for model predictions. **They raised two properties attention weights should hold if they provide faithful explanations:** (1) attention weights should correlate with feature importance-based methods such as gradient-based methods (Section 4.1); (2) altering attention weights ought to yield corresponding changes in model predictions. However, through extensive experiments in the context of text classification, question answer and natural language inference using a BiLSTM model with a standard attention mechanism, they found neither property is consistently satisfied: for the first property, they observed that attention weights do not provide strong agreements with the gradient-based approach [116] or the leave-one-out approach [118], as measured by the Kendall τ correlation; for the second property, by randomly re-assigning attention weights or generating adversarial attention distributions, [86] found that alternative attention weights do not essentially change the model outputs, indicating that the learned attention patterns cannot consistently provide transparent and faithful explanations. [164] reached a similar conclusion to [86] by examining intermediate representations to assess whether attention weights can provide consistent explanations. They used two ways to measure attention interpretability: (1) calculating the difference between two JS divergences – one coming from the model’s output distribution after zeroing out a set of the most attended tokens, and the other coming from the distribution after zeroing out a random set of tokens; (2) computing the fraction of decision flips caused by zeroing out the most attended tokens. Experiments demonstrate that higher attention weights has a larger impact on neither of these two measures.

Positive opinions come from [203, 193]. [203] challenged the opinion of [86] and argued that testing the attention’s interpretability needs to take into account all elements of the model, rather than solely the attention scores. To this end, [203] proposed to assess attention’s interpretability from four perspectives (an illustration is shown in Figure 9): (1) freeze the attention weights to be a uniform distribution and find that this variant performs as well as learned attention weights, indicating that the adversarial attention distributions proposed in [86] are not evidence against attention as explanation in such cases; (2) examine the expected variance induced by multiple training runs with different initialization seeds and find that the attention distributions seem not to be distanced much; (3) use a simple diagnostic tool which tests attention distributions by applying them as frozen weights in a non-contextual multi-layer perceptron model, and find that the model achieves better performance compared to a self-learned counterpart, demonstrating that attention indeed provides meaning model-agnostic explanations; (4) propose a model-consistent training protocol to produce adversarial attention weights and find that the resulting adversarial attention does not perform well in the diagnostic MLP setting. These findings correct some of the flaws in [86] and suggest that attention can provide plausible and faithful explanations.

[193] tested attention interpretability on a bulk of NLP tasks including text classification, pairwise text classification and text generation. Besides vanilla attention, self-attention is also considered for comparison. Through automatic evaluation and human

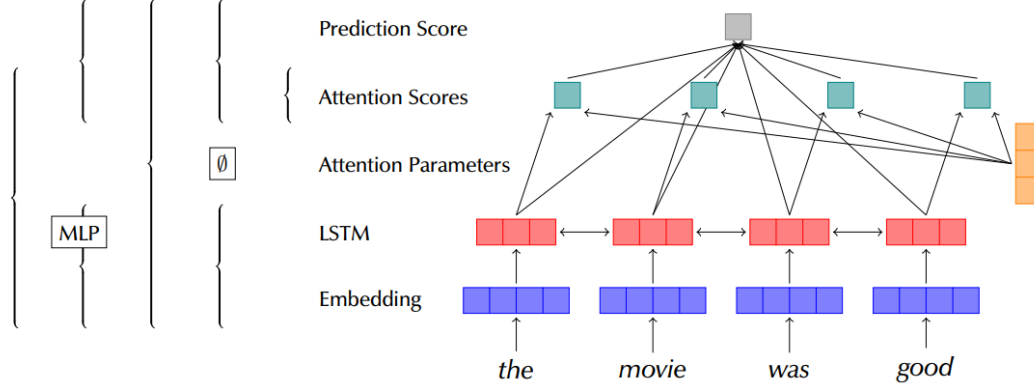


Figure 9: A schematic diagram of an LSTM model with attention (the figure is borrowed from [203]). [86] only manipulated attention at the *attention scores* level, while [203] manipulated attention at various levels and components of the model.

evaluation, [193] concluded that for tasks involving two sequences, attention weights are consistent with feature importance methods and human evaluation; but for single sequence tasks, attention weights do not provide faithful explanations regarding the model performance.

Upon the time of this review, whether the attention mechanism can be straightforwardly used for model interpretation is still debatable.

4.2.3 Attention for interpreting model decisions

Putting aside the ongoing debate, here we describe recently proposed attention-based interpretation methods. An early work comes from [201], who learned aspect embeddings to guide the model to make aspect-specific decisions. When integrated with different aspects, the learned attention shows different patterns for an input sentence. As an example shown in Figure 5 of the original literature, given the input “the appetizers are ok, but the service is slow”, when the `service` aspect embedding is integrated, the model attends to the content of “service is slow”, and when the `food` aspect embedding is integrated, the model attends to “appetizers are ok”, displaying that the attention can, at least to some degree, learn the most essential part of the input according to different aspects of interest.

For neural machine translation, during each decoding step, the model needs to attend to the most relevant part of the source sentence, and then translates it into the target language [15, 126]. [62] **investigated the word alignment induced by attention and compared the similarity of the induced word alignment to traditional word alignment**. They concluded that attention agrees with traditional alignment to some extent: consistent in some cases while not in other cases. [112] built an interactive visualization tool for manipulating attention in neural machine translation systems. This tool supports automatic and custom attention weights and visualizes output probabilities in accordance with the attention weights. This helps users to understand how attention affects model predictions. For natural language inference, [63] examined the attention weights and the attention saliency, illustrating that attention heatmaps identify the alignment evidence supporting model predictions, and the attention saliency shows how much such alignment impacts the decisions.

[123, 182] proposed self-interpreting structures that can improve model performance and interpretability at the same time. The key idea is to learn an attention distribution over the input itself (and may be at different granularity, e.g., the word level or the span level). Such attention distribution would be interpreted to some extent as evidence of how the model reasons about the given input with respect to the output. The core idea behind [123] is to learn an attention distribution over the input word-level features and average the input features according to the attention weights. They further extended the standalone attention distribution to multiple ones, each focusing on a different aspect and leading to a specific sentence embedding. This enables direct interpretation from the learned sentences embeddings. Suppose the input features are represented as a feature matrix $H \in \mathbb{R}^{n \times d_1}$ where n is input length and d_1 is the feature dimensionality. Two matrices $W_1 \in \mathbb{R}^{d_2 \times d_1}$ and $W_2 \in \mathbb{R}^{r \times d_2}$ are used to transform the feature matrix into attention logits, followed by a softmax operator to derive the final attention weights:

$$A = \text{softmax}(W_2 \tanh(W_1 H^\top)) \quad (19)$$

The resulting attention matrix A is of size $r \times n$, where each row is a unique attention distribution adhering to a particular

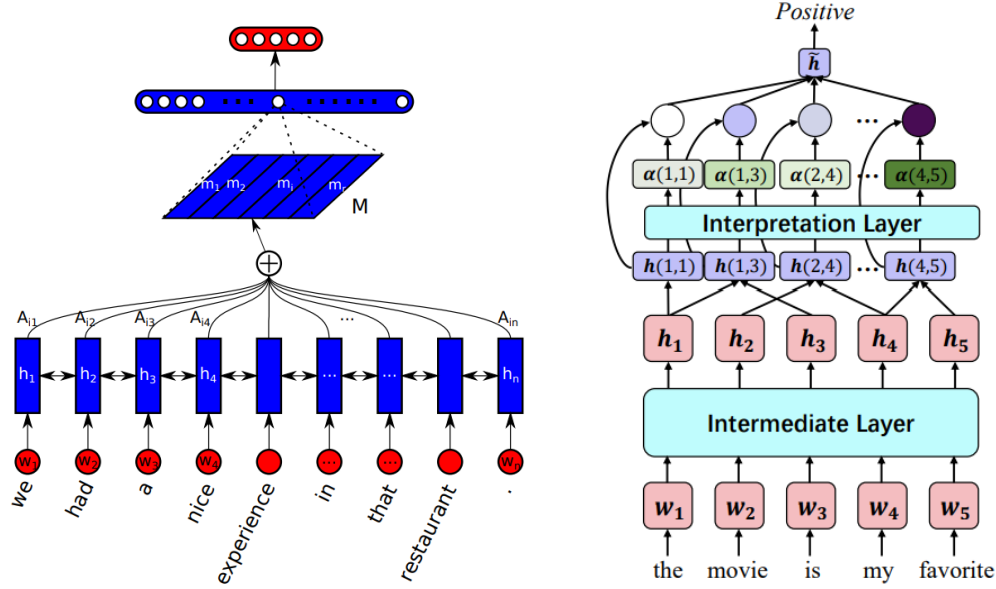


Figure 10: Diagrams of the self-attentive sentence embedding model [201] and attention-based self-explaining structure [182]. The figures are taken from the original literature.

aspect. Finally, the sentence embeddings $M \in \mathbb{R}^{r \times d_1}$ are obtained by applying A to the original feature matrix H :

$$M = AH \quad (20)$$

According to different output classes, the self-attentive model is able to capture different parts of the input responsible for the model prediction.

[182] extended this idea to the span-level, rather than just the word-level. They first represent each span by considering the start-index representation and the end-index representation of that span, forming a span-specific representation $h(i, j)$; then they treat these spans as basic units and compute an attention distribution in a way similar to [123]; last, the spans are averaged according to the learned attention weights for final model decision. This span-level strategy provides better flexibility to model interpretation. An overview of the self-interpreting structures proposed in [123] and [182] is shown in Figure 10.

4.2.4 BERT attention encoding linguistic notions

Built on top of the self-attention mechanism and Transformer blocks, BERT-style pretraining models [52, 210, 93, 125, 42, 39, 145, 29, 115, 177, 55, 183] have been used as the backbone across a wide range of downstream NLP tasks. The attention pattern behind BERT encodes meaning linguistic knowledge [154], which cannot be easily observed in LSTM based models.

BERT attention reveals part-of-speech patterns

[195] observed that BERT encodes part-of-speech (POS) tags. POS is a category of words that have similar properties and display similar syntactic behaviors. Commonly, English part-of-speeches include noun, verb, adjective, adverb, pronoun, preposition, conjunction, etc. [195] observed that the attention heads that focus on a particular POS tag tend to cluster by layer depth. For example, the top heads attending to proper nouns are all in the last three layers of the model, which may be due to that deep layers focus more on named entities; the top heads attending to determiners are all in the first four layers of the model, indicating that deeper layers focus on higher-level linguistic properties. [105] studied POS attention distributions on a variety of natural language understanding tasks. They found that the nouns are the most attended except the special $[\text{SEP}]$ token used for classification (as shown in Figure 11). These discoveries reveal that BERT can encode POS patterns through attention distributions, but may vary across layers and tasks.

BERT attention reveals dependency relation patterns

Dependency relation provides a principled way to structuralize a sentence’s syntactic properties. A dependency relation between a pair of words is a head-dependent relation, where a labeled arc links from the head to the dependent, representing their syntactic relationship. Dependency relations depict the fundamental syntax information of a sentence and are crucial for understanding

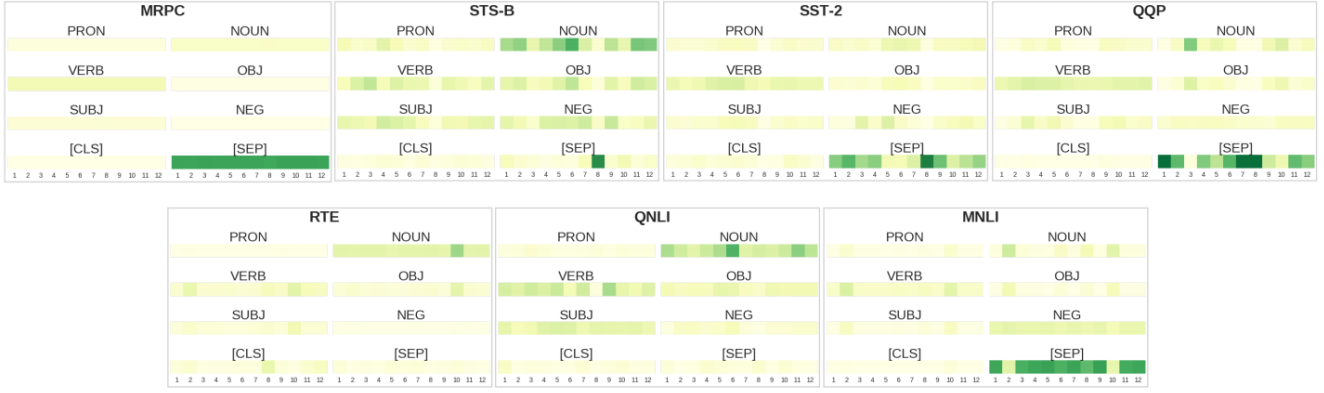


Figure 11: Per-task attention weights to part-of-speeches and the special [SEP] token averaged over input length and over the entire dataset. Except the [SEP], noun words are the most focused (the figure is brought from [105]).

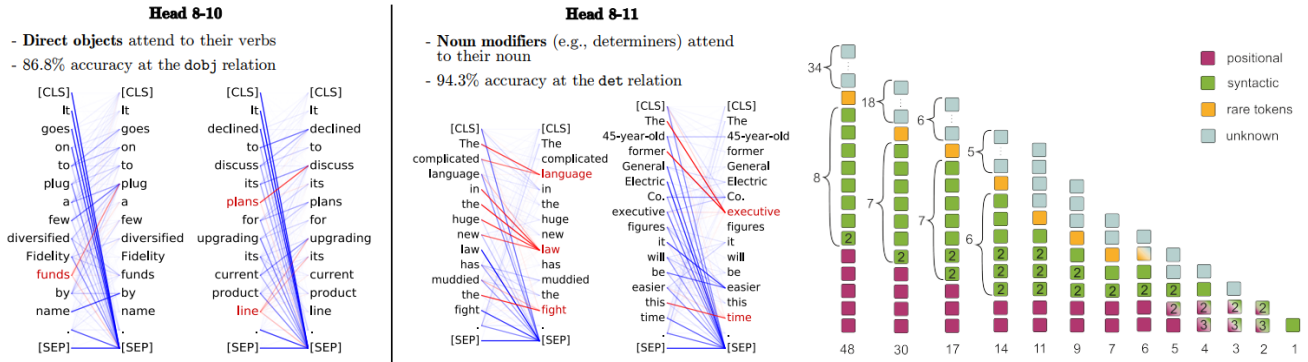


Figure 12: *Left*: Two examples of BERT attention heads that encode dependency relations (the figure is brought from [41]). *Right*: The functions of attention heads retrained after pruning (the figure is brought from [196]).

the semantics of that sentence. Extensive works show that BERT encodes dependency relations [66, 195, 41, 78, 122, 196, 150]. [66] found that BERT consistently assigns higher attention scores to the correct verb forms as opposed to the incorrect one in a masked language modeling task, suggesting some ability to model subject-verb agreement. Similar phenomena are observed by [122]. [41, 78] decoupled the “dependency relation encoding” ability from different heads and concluded that there is no single attention head that does well at syntax across all types of relations, and that certain attention heads specialize to specific dependency relations, achieving high accuracy than fixed-offset baselines. Figure 12 (left) shows several examples of dependency relations encoded in specialized attention heads. [196] investigated in depth the linguistic roles different heads play through attention patterns on the machine translation task. They found that only a small fraction of heads encode important linguistic features including dependency relations, and after pruning unimportant heads, the model still captures dependency relations in the rest heads and its performance does not significantly degrade. Figure 12 (right) shows the functions of heads retrained after pruning. [150] proposed *attention probe*, in which a *model-wide attention vector* is formed by concatenating the entries a_{ij} in every attention matrix from every attention head in every layer. Feeding this vector into a linear model for dependency relation classification, we are able to understand whether the attention patterns indeed encode dependency information. They had an accuracy of 85.8% for binary probe and an accuracy of 71.9% for multi-class probe, indicating that syntactic information is in fact encoded in the attention vectors.

BERT attention reveals negation scope

BERT encodes is the negation scope [220]. *Negation* is a grammatical structure that reverses the truth value of a proposition. The tokens that express the presence of negation are the *negation cue* such as word “no”, and the tokens that are covered in syntax by the negation cue are within the *negation scope*. [220] studied whether BERT pays attention to negation scope, i.e., the tokens within the negation scope are attending to the negation cue token. They found that before fine-tuning, several attention heads consistently encode negation scope knowledge, and outperform a fixed-offset baseline. After fine-tuning on a negation scope task, the average sensitivity of attention heads toward negation scope detection improves for all model variants. Their experiment results provide evidence for BERT’s ability of encoding negation scope.

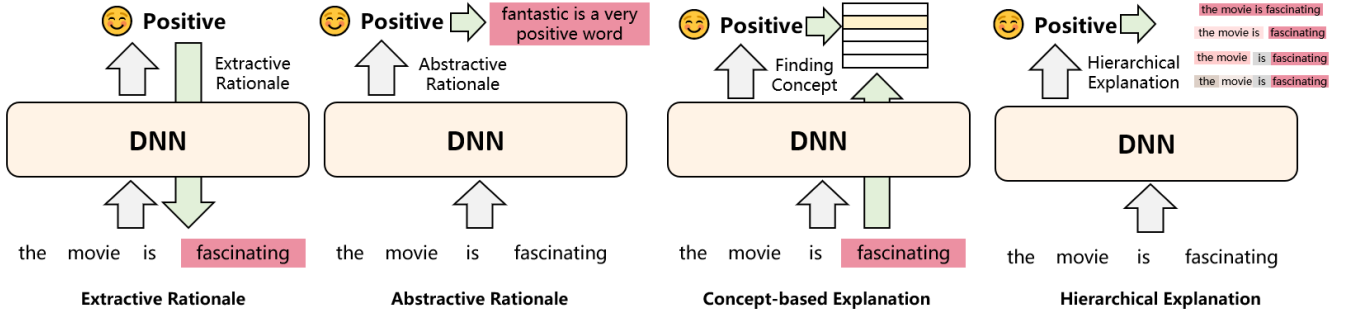


Figure 13: An overview of explanation generation methods. *First: extractive rationale generation* identifies rationales in the manner of extracting part(s) of the input. *Second: abstractive rationale generation* generates explanations in a sequence-to-sequence form by leveraging language models. *Third: Concept-based methods* interpret neural models by finding the most relevant concepts. *Fourth: Hierarchy-based methods* produce semantic hierarchies of the input features.

BERT attention reveals coreference patterns

BERT attention encodes semantic patterns, such as patterns for coreference resolution. Coreference resolution is the task of finding all mentions that refer to the same entity in text. For example, in the sentence “I voted for Nader because he was most aligned with my values”, the mentions “I” and “my” refer to the same entity, which is the speaker, and the mentions “Nader” and “he” both refer to the person “Nader”. Coreference resolution is a challenging semantic task because it usually requires longer semantic dependencies than syntactic relations. [41] computed the percentage of the times the head word of a coreferent mention attends to the head of one that mention’s antecedents. They find that attention achieves decent performances, attaining an improvement by over 10 accuracy points compared to a string-matching baseline and performing on par with a rule-based model. [122] explored the reflexive anaphora knowledge attention encodes. They defined a *confusion score*, which is the binary cross entropy of the normalized attention distribution between the anaphor and its candidate antecedents. They find that BERT attention does indeed encode some kind of coreference relationships that render the model to preferentially attend to the correct antecedent, though attention does not necessarily correlate with the linguistic effects we find in natural language.

Since the attention mechanism is nested in and jointly trained with the main model, interpretation methods based on attentions thus belong to the *joint* line.

4.3 Explanation Generation Based Interpretation

Saliency maps and attention distributions assign importance scores, which cannot be immediately transformed to human-interpretable explanations. This issue is referred to as *explanation indeterminacy*. Explanation-based methods attempt to address this problem by generating explanations in the form of texts.

Explanation-based methods fall into three major categories: *rationale-based* explanation generation, *concept-based* explanation generation and *hierarchical* explanation generation. Rationale-based explanation generation is further sub-categorized into *extractive* rationale generation and *abstractive* rationale generation, which respectively refer to generating rationales within the input text in an extractive fashion, and generating rationales in a sequence-to-sequence model in a generative fashion. Concept-based methods aim at identifying the most relevant concepts in the concept space, where each concept groups a set of examples that share the same meaning as that concept: **a concept represents a high-level common semantics over various instances**. Hierarchy-based methods construct a hierarchy of input features and provide interpretation by examining the contribution of each identified phrase. Explanation-based methods directly produce human-interpretable explanations in the form of texts, and thus are more straightforward to comprehend and easier to use.

Figure 13 provides a high-level overview for the four categories of generation-based methods.

4.3.1 Rationale-based explanation generation

A rationale is defined to be a short yet sufficient piece of text serving as justification for the model prediction [113]. A *good* rationale should conceptually satisfy the following properties [212]:

- *Sufficiency*: If the original complete input x leads to the correct model prediction, then using only the rationale should also induce the same prediction.
- *Comprehensiveness*: The non-rationale counterpart should not contain sufficient information to predict the correct label.

<p>a beer that is not sold in my neck of the woods , but managed to get while on a roadtrip . poured into an imperial pint glass with a generous head that sustained life throughout . nothing out of the ordinary here , but a good brew still . body was kind of heavy , but not thick . the hop smell was excellent and enticing . very drinkable</p>
<p>very dark beer . pours a nice finger and a half of creamy foam and stays throughout the beer . smells of coffee and roasted malt . has a major coffee-like taste with hints of chocolate . if you like black coffee , you will love this porter . creamy smooth mouthfeel and definitely gets smoother on the palate once it warms . it 's an ok porter but i feel there are much better one 's out there .</p>
<p>i really did not like this . it just seemed extremely watery . i dont ' think this had any carbonation whatsoever . maybe it was flat , who knows ? but even if i got a bad brew i do n't see how this would possibly be something i 'd get time and time again . i could taste the hops towards the middle , but the beer got pretty nasty towards the bottom . i would never drink this again , unless it was free . i 'm kind of upset i bought this .</p>
<p>a : poured a nice dark brown with a tan colored head about half an inch thick , nice red/garnet accents when held to the light . little clumps of lacing all around the glass , not too shabby . not terribly impressive though s : smells like a more guinness-y guinness really , there are some roasted malts there , signature guinness smells , less burnt though , a little bit of chocolate ... m : relatively thick , it is n't an export stout or imperial stout , but still is pretty hefty in the mouth , very smooth , not much carbonation . not too shabby d : not quite as drinkable as the draught , but still not too bad . i could easily see drinking a few of these .</p>

Figure 14: Examples of extracted rationales indicating the sentiments of various aspects (appearance:red, smell: blue, palate: green). This figure is from [113].

- *Compactness*: The segments of the original input text that are included in the rationales should be sparse and consecutive, i.e., rationales should be as concise as possible.

We describe two styles of rationale generation, *extractive* and *abstractive*, and review representative works in the rest of this section.

Extractive rationale generation

[113] propose to extract rationales as interpretable justifications for the model prediction. The rationale generation process is incorporated as an integral part the overall learning problem, where the model consists of two modules: the *generator*, which specifies a distribution over possible rationales, and the *encoder*, which maps any text into target values. The generator and the encoder are jointly trained to optimize the objective that favors short, compact and sufficient rationales while preserving accurate predictions. More concretely, the encoder is used to digest the given input and output the predicted label. The encoder can be trained using a standard loss function, say, the squared error:

$$\mathcal{L}_{\text{enc}}(x, y) = \|\text{enc}(x) - y\|_2^2 \quad (21)$$

At the meantime, a generator is employed to extract a subset of the original input text x serving as the rationale. [113] used binary variables $\{z_1, \dots, z_i, \dots, z_l\}$ where l is the length of the input x to denote whether each word x_i should be selected or not. This process can be formalized as:

$$z \sim \text{gen}(x) \triangleq p(z|x) = \prod_{i=1}^l p(z_i|x, z_{<i}) \quad (22)$$

Equation 22 indicates that the binary variable z_i at time step i is determined by the input x as well as the previously generated binary variables $z_{<i}$, making the generation process sequentially dependent. With the extracted rationales at hand, we can rewrite Equation 21 into the following form:

$$\mathcal{L}_{\text{enc}}(z, x, y) = \|\text{enc}(z, x) - y\|_2^2 \quad (23)$$

where $\text{enc}(z, x)$ means that the encoder is only applied to the rationales rather than the entire input text. Equation 23 encourages sufficiency of the extracted rationales. Because we would like the extracted rationales to be as concise and compact as possible, we enforce the generator minimize the following regularizer:

$$\mathcal{L}_{\text{gen}}(z) = \lambda_1 \|z\| + \lambda_2 \sum_i |z_i - z_{i-1}| \quad (24)$$

Improvement	Representative works
Decoupling rationale selection and label prediction.	[87]
Improving masking strategies.	[19, 50]
Extracting rationales according to or conditioning on different classes.	[33, 212]
Incorporating the information bottleneck theory.	[138]
Extracting rationales through matching.	[185, 79]

Table 4: Improvements of extractive explanation generation over [113].

The first term encourages conciseness and the second term encourages consecutiveness. Overall, Equation 24 compactness of the extracted rationales.

Combining Equation 23 and Equation 24 together leads to the final training objective:

$$\min_{\theta_{\text{enc}}, \theta_{\text{gen}}} \sum_{(x, y)} \mathbb{E}_{z \sim \text{gen}(x)} [\mathcal{L}_{\text{enc}}(z, x, y) + \mathcal{L}_{\text{gen}}(z)] \quad (25)$$

Equation 25 is optimized by doubly stochastic gradient descent, a variant of the REINFORCE algorithm [205] as a sampled approximation to the gradient of Equation 25. Figure 14 shows several examples of the extracted rationales associated with the sentiments of different aspects. The proposed method successfully extracts interpretable rationales that can sufficiently explain the model predictions, and the extraction text snippets are generally concise and compact.

A large strand of works are motivated by [113] and propose to improve extractive rationale generation from various perspectives. We introduce some typical directions in the following contents. A summary of these improvements is provided in Table 4.

Decoupling rationale selection and label prediction [87]. A drawback of the vanilla extractive rationale generation approach is the difficulty of training the two components – the encoder and the generator simultaneously using only instance-level supervision, as shown in Equation 25. To tackle this issue, [87] proposed to decouple the rationale selection process and the label prediction process, separately training the two modules and respectively applying them for rationale extraction and label prediction. The proposed pipeline works as follows: (1) train a support model to assign importance scores (Section 4.1 and Section 4.2.1) to input features, and discretize them into binary labels similar to the latent binary labels z in [113]; (2) treating the original text as input and the binarized labels as gold output, train an extraction model to extract rationales; (3) treating the extracted rationales as input and the gold class label as output, train a classifier for label prediction. The training pipeline introduces independent modules for rationale extraction and label prediction, naturally bypassing heavy recourse to reinforcement learning and mitigating the training difficulty.

Improving masking strategies [19, 50]. This line of methods improve the masking strategy, i.e., how to assign binary variables as in [113] to avoid sophisticated model training. The basic idea is to impose differentiability on the binary latent variables and the sparsity-inducing regularization, making solving Equation 25 tractable even without REINFORCE. [19] proposed to use the Kumaraswamy distribution [108], a family of distributions that exhibit both discreteness and continuity, to allow for reparameterized gradient estimates and support for binary outcomes. [50] proposed to learn to mask out subsets while maintaining differentiability by pushing close the output predicted using the original input text and the output predicted using the extracted rationales. An overview of the proposed model in [50] is shown in Figure 15 (left).

Extracting rationales according to or conditioning on different classes [33, 212]. The common paradigm we have introduced above is to make an *overall* selection of a subset of the input that maximally explains the model decision, but rationales can be *multi-faceted* regarding different classes. To tackle this issue, [33] proposed *class-wise rationales*, where different sets of rationales are generated to support the decisions on different classes. [212] similarly proposed to generate class-wise rationales, and more importantly they explicitly control the rationale complement via an adversarial model so as not to leave any useful information out of the rationale selection. The fine-grained control of rationale complement remedies the comprehensiveness property missed by the vanilla extractive model. An overview of the model proposed by [212] is shown in Figure 15 (right).

Incorporating the information bottleneck theory [37, 138]. The reduction of the original input text inevitably results in performance degradation because the rationale complement usually contains non-essential but useful information. The trade-off between concise explanations (rationales) and high task accuracy can be better managed by incorporating the information bottleneck objective, where a rationale is expected to be (1) minimally informative about the original input and (2) maximally informative about the output class. A prior distribution provides flexible controls over the sparsity level, i.e., the fraction of the extracted rationales among the entire input text. Figure 16 (left) provides an overview of this method.

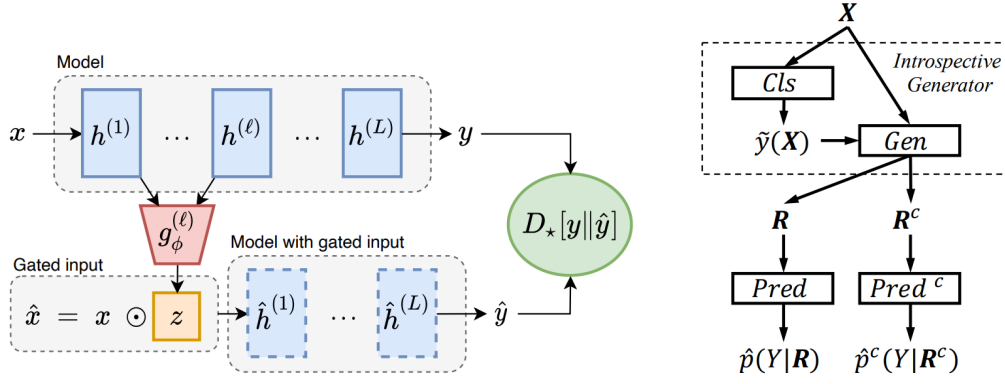


Figure 15: Diagrams of the differentiable masking strategy (left, the figure is brought from [50]) and class-dependent method (right, the figure is brought from [212]). *Left*: Hidden states up to layer ℓ are fed to a classifier to predict the mask, which is then used to run a forward pass with the input masked. The objective is to minimize the divergence. *Right*: An introspective generator first predicts the possible output \hat{y} and then generates rationales and rationale complements based on x and \hat{y} .

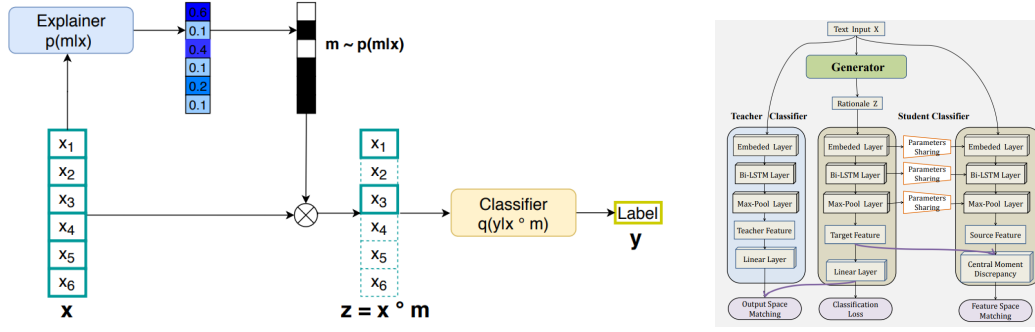


Figure 16: Diagrams of the information bottleneck enhanced model (left, the figure is brought from [138]) and the distribution matching model (right, the figure is brought from [79]). *Left*: An explainer first extracts rationales from the input and then a classifier predicts the output based only on the rationale. The explainer leverages the Gumbel-Softmax reparameterization [88] to reparameterize the Bernoulli variables. The full model is optimized by an information bottleneck guided objective. *Right*: The generator is enforced to generate rationales that match the full input feature in the feature space and match the full input prediction in the output space.

Extracting rationales through matching [185, 79]. Intuitively, similar inputs should have similar rationales, and the extracted rationales should also get close to the original input in the feature space. Motivated by this intuition, better rationales can be extracted by matching similar inputs, or matching the distributions in the latent space. [185] proposed to employ Optimal Transport (OT) [49] to find a minimal cost alignment between a pair of inputs, providing a mathematical justification for rationale selection. [79] did not perform matching in the textual level, but rather in the feature space and the output space. In the feature space, they imposed a regularizer that minimizes the central moment discrepancy [213] between the full input feature and the rationale feature. In the output space, they minimized the cross entropy loss between the full input prediction and the rationale prediction, acting in a way similar to knowledge distillation [76]. Figure 16 (right) shows the method of [79].

Abstractive rationale generation

The extractive style rationale generation limits the space of rationales to the segments of the input text. In some cases, we would like the rationale to reflect the reasoning process of the model decision. For example, when predicting the input “the movie is fantastic” as label `positive`, the model should be able to reason that “fantastic is a very positive word, and there is no negation that pushes the sentiment to the other side”. Extractive rationale generation would only, however, highlight the word “fantastic”, but cannot explain why to select that word. In such cases, the model needs to provide *free-text* rationales by means of generating from scratch based on the input and the output. This type of rationale generation is called *abstractive* rationale generation, relative to *extractive* rationale generation.

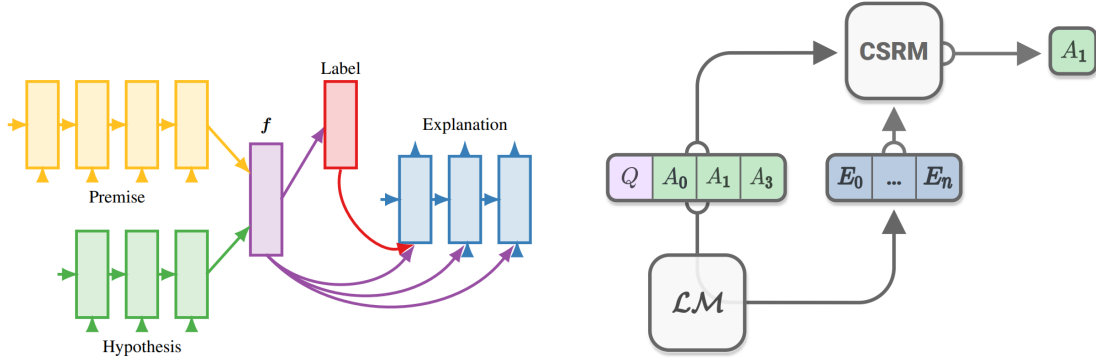


Figure 17: Diagrams of the e-InferSent architecture (left, the figure is brought from [31]) and the CSRM (Commonsense Reasoning Model) (right, the figure is brought from [149]). *Left*: Given a pair of premise and hypothesis, the e-InferSent architecture jointly models the label and the corresponding explanation. *Right*: A language model is first used to generate explanations based on the question Q and candidate answers $\{A_0, \dots, A_n\}$, and CSRM is then used to predict the right answer.

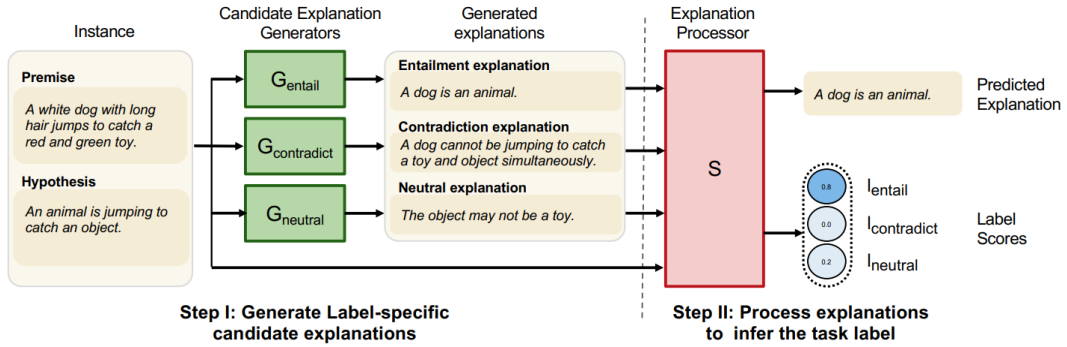


Figure 18: The workflow of the proposed model in [107]. Given a pair of premise and hypothesis, the generator first produces label-specific explanations, which are then fed into the processor to infer label scores using the evidence present in these explanations. The figure is taken from the original literature.

The abstractive style rationale generation is usually implemented by natural language generation models, e.g., language models and sequence-to-sequence models. A common practice is to first craft datasets including the input instances as well as the corresponding human-annotated rationales, and then train language models in a supervised manner. For example, [31] crafted the e-SNLI dataset, which is an extension of SNLI [26] focusing on providing human-labeled explanations for entailment relations; [149] created CoS-E (Common Sense Explanations), an extension of CommonsenseQA [189], providing human explanations for commonsense reasoning. Both datasets offer direct supervision for model training, endowing the generation model with the ability to reason about its prediction and produce free-text rationales. Figure 17 shows the model architectures proposed in [31] and [149].

Similar to extractive rationale generation, the abstractive model can also generate label-specific explanations and then examine the faithfulness of the generated explanation with respect to the corresponding label [124, 107, 202]. A typical work comes from [107], the workflow of which is depicted in Figure 18. A merit that label-specific explanations offer is that it allows for re-examination on the generated rationales and on the whether these rationales can faithfully interpret the labels. **This rationale generation approach supports testable explanations of the decisions and improves classification accuracy, even with limited amounts of labeled data.**

An important advantage of abstractive rationale generation over extractive rationale generation is that abstractive models can be enhanced by large-scale language model pretraining [146, 134]. Pretrained on massive unlabeled general texts and finetuned on labeled rationale-guided data, the model is able to produce more coherent and human-identifiable explanations.

Extractive style and abstractive style rationale generation can be bridged via commonsense knowledge, rendering both types of explanations better [128]. The outlined pipeline works as follows: (1) extracting rationales most responsible for the prediction (using extractive rationale generation methods); (2) expanding the extracted rationales using commonsense resources; (3) using the expanded knowledge to generate free-text explanations. This pipeline naturally infuses commonsense knowledge into the

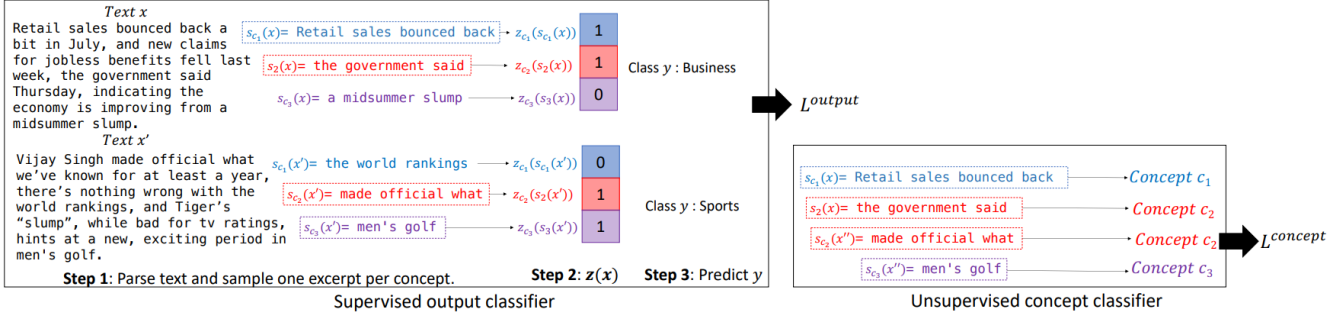


Figure 19: The workflow of the proposed EDUCE model in [25]. The model first samples one excerpt per concept, then determines whether the excerpt really activates the concept, and last feeds the binary representation for classification. An unsupervised concept classifier is used to enforce concepts consistency and to prevent overlap of concepts. The figure is taken from the original literature.

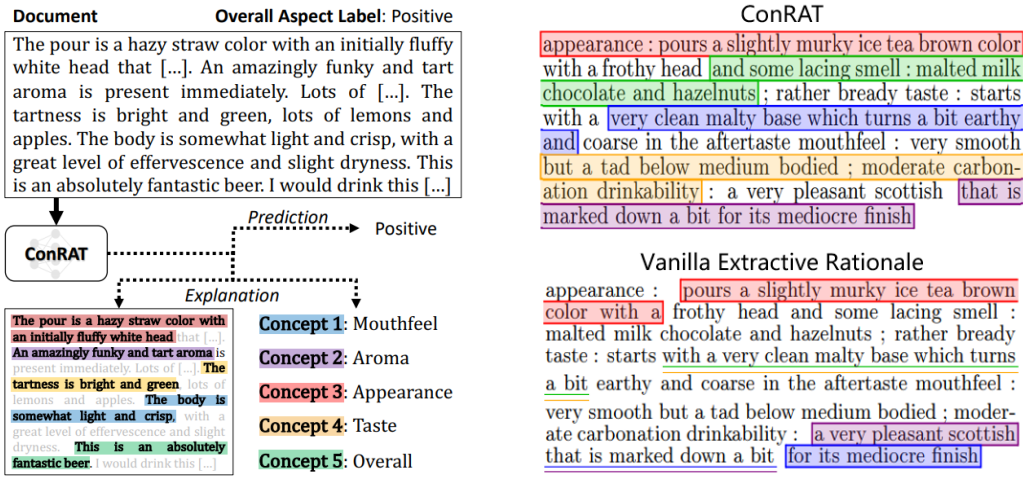


Figure 20: *Left*: The pipeline of the proposed model ConRAT in [7]. ConRAT first extracts a set of text snippets from the input and infers which ones are described as concepts (excerpts). Then the model explains the outcome with a linear aggregation of the concepts. *Right*: A comparison of the outputs produced by ConRAT and vanilla extractive rationale generation [113]. The contents marked in red, green, yellow, blue and purple are respectively for appearance, aroma, palate, taste and overall sentiment. Vanilla extractive rationale produces spurious correlations while ConRAT extracts meaningful and non-overlapping concepts. The figures are taken from the original literature.

process of rationale generation, therefore benefiting generating explanations more understandable to humans.

4.3.2 Concept-based explanation generation

Following the definition in [18], a *concept* [5, 119, 98] is represented as a vector in the space which groups all the set of examples (rationales) that share the meaning of the concept. If the extracted rationale corresponds to a specific concept, that rationale is called an *excerpt* [25]. For example, the phrases of “fantastic movie”, “enjoy the movie” and “the acting is very good” share the same general concept *positive*, and all of these three phrases are excerpts that trigger the *positive* concept. **One benefit provided by concept-based methods is the high-level clusterings of rationales that share the same general semantics.** A concept can be simply the class label [18], e.g., the negative concept and the positive concept; a concept can also be more complex forms such as sub-categories of a class label [25]. For example, ask a user to classify the given text “Retail sales bounced back a bit in July, and new claims for jobless benefits fell last week, the government said.” The user could detect “*retail sales*” and, say, recognize it an *economy* concept; she would also note “*the government said*” and recognize it as an *politics* concept. Combining both concepts, the user can classify this input as label *Business*. If the model correctly predicts the label *Business*, and at the same time extracts text pieces as well as their corresponding concepts, the model prediction would be more interpretable. Extractive and abstractive rationale generations, however, cannot provide such high-level explanations, but only aim at interpret individual-specific inputs.

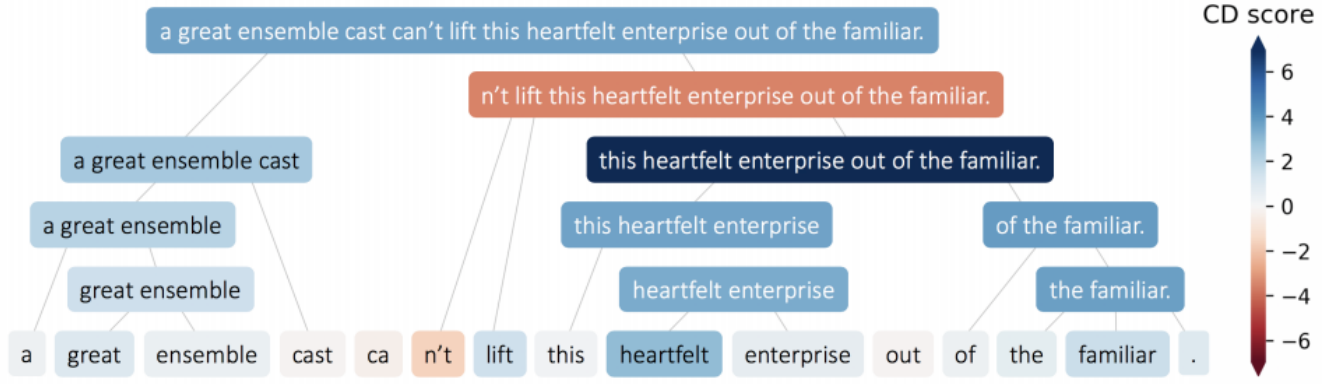


Figure 21: An example from [173] illustrating how the explanation hierarchy works. Blue represents positive sentiment, white is neutral and red is negative. Higher rows display phrases associated with their sentiment scores identifies by the model.

Existing literature generates concepts in an unsupervised manner [25, 18, 147, 7], where the concept is selected without any human annotation or priori. Two typical works come from [25] and [7]. [25] proposed EDUCE (Explaining model Decisions through Unsupervised Concepts Extraction), which works in the following steps: (1) Compute $p(s|x, c)$, the probability of each excerpt s in the input x for each concept c , and sample one excerpt $s_c(x)$ according to the probability; (2) For each concept, determine whether the extracted excerpt is actually the trigger, i.e. computing $p(z_c|s_c(x), c)$. $z_c(s_c(x)) = 1$ means concept c is detected as present, and absent otherwise; (3) Predict the class label y using only the binary concept vector $z(x) = (z_1(s_1(x)), \dots, z_C(s_C(x)))$, where C is the number of concepts. In order for semantic consistency and to avoid overlap of the concepts, [25] jointly trained a concept classifier to categorize each excerpt into one of the C classes, which exactly performs like a multi-label classifier. An overview is provided in Figure 19.

[7] proposed ConRAT, short for Concept-based Rationalizer, which could achieve stronger performance over [25] and infer non-overlapping and human-understandable concepts. The ConRAT model consists of two modules: a generator that finds the concepts, and a selector that detects whether a concept is present or absent. The key difference from [25] lies in the way of forming the concepts: [25] employed a simple concept classifier to decouple different concepts, whereas [7] (1) favors the orthogonality of concepts and (2) minimizes the cosine similarity between concepts. Combining both regularizers, ConRAT is shown to produce accurate and interpretable concepts. Figure 20 (left) illustrates the pipeline of ConRAT. Figure 20 (right) compares the rationales output by ConRAT and the vanilla extractive rationale generation method [113].

4.3.3 Hierarchical explanation generation

The third family of generation-based methods [133, 173, 92, 36] interpret model predictions by decomposing the input into tree-like segments at different granularity and computing the contribution of each segment to the model prediction. This forms a contextual hierarchy from the bottom level of words to the high level of phrases, enabling intuitive observations on how different parts of the input combine with each other and how their semantics influence the model prediction. Compared to rationale-based and concept-based methods, hierarchical explanation generation offers the most fine-grained interpretation, as they quantitatively measure the contribution of each detected phrase to the model prediction and exhibit the interpretable interactions between phrases.

The pioneer works [133, 173] for hierarchical explanation generation proposed *Contextual Decomposition* (CD), which decomposes the input into a hierarchy of phrases and assigns relevance scores signifying the contribution of each phrases to the model prediction. An example is shown in Figure 21. Concretely, a neural network $f(x)$ can be formalized as a composition of functions, followed by a softmax operator to derive the final prediction:

$$f(x) = \text{softmax}(g(x)) = \text{softmax}(g_L(g_{L-1}(\dots(g_1(x)))))) \quad (26)$$

where $g(x)$ is the logits produced by the neural network. The logits, however, can be decomposed into the addition of two terms, $\beta(x)$ and $\gamma(x)$, where $\beta(x)$ captures the importance of the each input feature x_i and $\gamma(x)$ captures the residue:

$$g_i(x) = \beta_i(x) + \gamma_i(x), \forall i = 1, 2, \dots, L \quad (27)$$

The remaining question is how to compute $\beta_i(x)$ and $\gamma_i(x)$ for different neural architectures. [133] proposed to disambiguate the interactions between gates (i.e. the output gate o_t , the forget gate f_t and the input gate i_t) in LSTMs [77] by linearizing the

update function. [173] extended this idea to general DNN structures including convolutional, max-pooling, ReLU and dropout layers, etc. Besides, [133] introduced the idea of *hierarchical saliency*, where a cluster-level importance measure is used to agglomeratively group features the model learns as predictive. The model is thus able to generate interpretable hierarchies more compliant to the model prediction.

[92] pointed out two properties CD-based methods should satisfy: (1) *Non-additivity*: the importance of a phrase should not simply be the sum of the importance scores of all the component words; (2) *Context independence*: the importance of a phrase should be evaluated independent of its context. Unfortunately, neither the methods of [133] and [133] satisfy both properties. To address this limitation, [92] proposed a new importance measure that satisfies both the non-additivity and context independence properties, producing consistently better explanations.

[36] designed a top-down model-agnostic method of constructing hierarchical explanations via feature interaction detection. Starting from the full input sentence, the model recursively divides the segments into smaller parts, until each segment contains only one word. [36] defined a *partition score*, which represents the degree of interaction between two successive segments. A small partition score means that the two segments are weakly interacted, i.e., partitioning these two segments would not cause a drastic semantic shift regarding interpretation. To evaluate the contribution of each new segment, [36] further defined a *feature importance score*, which is implemented as the difference between the predicted probability of the ground-truth label and the maximum predicted probability among other labels, given only the segment itself as input. This top-down recursive procedure can be applied to any neural model architecture and is free from complex mathematical derivation.

Though some explanation generation based methods can use post-hoc approaches to produce interpretation, e.g., saliency scores can be computed to extract the rationale, most existing works jointly train the interpreting model and the main model. We thus categorize explanation generation based interpretation into the *joint* line.

4.4 Miscellaneous

It is noteworthy that there are many other important miscellaneous works we do not mention in the previous sections. For example, numerous works have proposed to improve upon vanilla gradient-based methods [174, 178, 65]; linguistic rules such as negation, morphological inflection can be extracted by neural models [141, 142, 158]; probing tasks can be used to explore linguistic properties of sentences [3, 80, 43, 75, 89, 74, 34]; the hidden state dynamics in recurrent nets are analysed to illuminate the learned long-range dependencies [73, 96, 67, 179, 94]; [169, 166, 168, 101, 57, 167] studied the ability of neural sequence models to induce lexical, grammatical and syntactic structures; [91, 90, 12, 136, 159, 24, 151, 85] modeled the reasoning process of the model to explain model behaviors; [157, 139, 28, 163, 219, 170, 180, 137, 106, 58, 162, 81, 111, 23] proposed to interpret word embeddings so that each dimension corresponds to a fine-grained sense and the value of that dimension represents the relevance of the sense to the word; [1] drew on external knowledge base to derive a complete reasoning sequence from the natural language utterance to the final answer; [192] interpreted neural systems by combining natural language and the human brain imaging recordings. We kindly refer the readers to the original literature of interest for details.

5 Hybrid Methods

Hybrid interpreting methods blend training-based interpretation and test-based interpretation, providing model interpretations from a global perspective. A typical work comes from [129], which jointly examines training history and test stimuli by associating influence functions and model gradients.

The first question hybrid methods answer is *which part(s) of the input example contribute most to the model prediction*. This question can be simply resolved by the techniques introduced in Section 4. However, as we would further like to interpret the detected part(s) from the training side, it is preferable to employ a differentiable method with which we can directly compute the influence functions regarding each training instance and the saliency part. [129] uses the *Vanilla Gradient* form for simplicity, and other gradient-based methods such as *Integrated Gradient* are also applicable. The saliency score for each input token x_i can be expressed as follows:

$$\text{Saliency}(x_i) = \|w_y(x_i)\|, \quad w_y(x_i) = \frac{\partial S_y(x)}{\partial x_i} \quad (28)$$

where x_i is the corresponding word embedding of that word. Recall in Equation 4 that the influence function $I(z, x)$ measures the influence of a training point z on an input instance x , while in this case, we would like to associate a training point z with a part of the input, say x_i . To this end, we can substitute x in Equation 4 with the partial derivative $w_y(x_i)$ we just computed:

$$I(z, w_y(x_i)) = -\nabla_{\theta} w_y(x_i)^{\top} H_{\theta}^{-1} \nabla_{\theta} L(z; \hat{\theta}) \quad (29)$$

TreeLSTM	BERT
<i>Test example 1: i loved it !</i>	
(a) if you 're a fan of the series you 'll love it.	(a) i loved this film .
(b) old people will love this movie.	(b) you 'll probably love it .
(c) ken russell would love this .	(c) a movie i loved on first sight
(d) i loved this film .	(d) old people will love this movie.
(e) an ideal love story for those intolerant of the more common saccharine genre .	(e) i like this movie a lot !
<i>Test example 2: it 's not life-affirming – its vulgar and mean , but i liked it .</i>	
(a) i like it .	(a) as an introduction to the man 's theories and influence , derrida is all but useless ; as a portrait of the artist as an endlessly inquisitive old man , however , it 's invaluable
(b) the more you think about the movie , the more you will probably like it .	(b) i like it .
(c) one of the best , most understated performances of jack nicholson 's career	(c) i liked the movie , but i know i would have liked it more if it had just gone that one step further .
(d) it 's not nearly as fresh or enjoyable as its predecessor , but there are enough high points to keep this from being a complete waste of time .	(d) sillier , cuter , and shorter than the first as best i remember , but still a very good time at the cinema
(e) i liked it just enough .	(e) too daft by half ... but supremely good natured .

Table 5: An exmaple drawn from [129]. The most salient token in the test example regarding the golden label y is marked in **red**. Each test example is paired with top 5 training examples that are the most responsible for the salient region by $I(z, w_y(x_i))$ in Equation 29. For each extracted training example, its constituent token that is the most responsible for the salient region by $I(z_t, w_y(x_i))$ in Equation 31 is marked in **green**.

Now for a given salient token x_i , Equation 29 quantitatively measures the contribution of each training point z to the detected part of the test input, given by $I(z, w_y(x_i))$. Moreover, the influence of perturbing a training example z into \tilde{z} on the salient part of a test example x can also be measured:

$$I(z, \tilde{z}, w_y(x_i)) \triangleq I(\tilde{z}, w_y(x_i)) - I(z, w_y(x_i)) = -\nabla_{\theta} w_y(x_i)^{\top} H_{\theta}^{-1} (\nabla_{\theta} L(\tilde{z}; \hat{\theta}) - \nabla_{\theta} L(z; \hat{\theta})) \quad (30)$$

where \tilde{z} is a perturbed version of the original training point z . When we implement perturbation as simple token removal, we can immediately quantify the influence of a certain token z_t of a training point z on the salience part $w_y(x_i)$:

$$I(z_t, w_y(x_i)) = I(\tilde{z}, w_y(x_i)) - I(z \setminus \{z_t\}, w_y(x_i)) \quad (31)$$

Jointly examining training history and test stimuli offers a more comprehensive perspective for interpreting model decisions. Table 5 shows an example drawn from [129] illustrating the identified salient part of the test input, the identified training points and the corresponding salient regions of these training points. Both models of TreeLSTM [186] and BERT [52] successfully detect the salient part (i.e. “loved” and “liked”) as well as the influential training points. [129] further applied the hybrid method to adversarial generation and model prediction fixing, exhibiting a wide utility of hybrid methods.

Hybrid methods combine influence functions and saliency scores and can be categorized into the *post-hoc* line.

6 Open Problems and Future Directions

In spite of the variety of interpreting methods proposed for neural NLP, quite a few open problems still remain:

- What interpretability exactly means and how to define interpretability: existing works aim at proposing new and more effective interpreting models, but few clarifies what exactly interpretability is, what the goal of interpretability is and how what philosophy we can take in general to reach interpretability [131]. One can “easily” state that interpretable models are those that produce human-understandable results with reasoning behind predictions explicitly captured by humans. But quite a few questions arise: how we can formally define “human-understandable”? What if the produced logic can not be recognized by some people but can be comprehended by others? The lack of a clear psychological definition and a formal mathematical formulation harms the development of further researches in neural NLP interpretation.

- How to evaluate interpreting methods: The lack of formal definition for interpretability leads to a lack of a widely accepted criterion for evaluation. Interpretability methods are currently evaluated by humans [72, 83, 202], which are inevitably subjective: For example, to evaluate saliency-based methods, we can visualize the saliency maps generated by different interpreting methods to see whether they correctly identify the responsible part of the input for the model prediction; we can plot the attention distributions to interpret model decisions; we can also let the model to generate explanations. However, there is no such standardized metrics used to automatically evaluate model interpretability. Some datasets such as ERASER [53] have defined automatically metrics, but they still require human-labeled data for evaluation. A robust and convenient criterion for the evaluation is urgently needed to compare the effectiveness of different interpretability models.
- Whose interpretability are we talking about? Different populations or users’ goals might affect which types of interpretability to use. In certain cases, end users or model practitioners may prefer different aspects of interpretability that may not need to be related to the task itself. We urge future work on interpretability to take into account more user centered design when developing interpretability methods and evaluations.
- Go beyond classification tasks. Existing interpretability methods are mainly designed for text classification methods. How can we extend the aforementioned interpretability methods to other tasks such as named entity recognition, text generation or summarization? Certain interpretability methods such as attention-based interpretation might still be applicable to some extent, however, saliency-based ones may be fragile as it is hard to connect a specific token with the entire text sequence.
- The inconsistency between different interpreting methods: It is common that existing interpreting methods contradict with each other when giving explanations, as shown in Figure 7. The inconsistency of different interpreting methods makes users bewildered in terms of interpreting a model’s decisions. Future directions should explore the relations between different perspectives that different interpreting methods take, the factors that make them diverge, and the way of unifying them.
- The cost of model performance in exchange for interpretability: Neural interpretation aims at interpreting neural models while preserving model performance, complexity or resource consumption [182]. Some of existing methods improve model interpretability, but jeopardize model performance [21]. Studying such trade-offs between interpretability and model characteristics such as model performance is crucial for understanding the relationship between interpretability and model behaviors.
- The utility of interpretability: The most important goal of developing interpretability methods is to use them, improving a neural model’s trustworthiness and controllability, helping improve and debug neural models, and facilitating the human AI collaboration. Currently, there hasn’t been a systematic paradigm under which an interpretability model can be used in practice. Future directions should explore the paradigm of how an interpretability model can be paired with its original model, where AI systems provide not only decisions but also the human-interpretable reasoning process, helping users to judge how much they can trust a result given by an AI system [135, 60, 109, 215, 17, 84].

7 Conclusion

This survey showcases recent advances of research in interpreting neural NLP models. We first introduce the high-level taxonomy of existing interpreting methods. We have categorized related literature from two perspectives: training-based v.s. test-based, and joint v.s. post-hoc. Then, we introduce typical works of each category in detail and shed light on their difference on the basic tools they use to interpret models. Last, we raise some open problems and future directions for this area in the hope of providing insights encouraging the community for a deeper understanding toward neural NLP interpretation.

References

- [1] ABUJABAL, A., ROY, R. S., YAHYA, M., AND WEIKUM, G. Quint: Interpretable question answering over knowledge bases. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* (2017), pp. 61–66.
- [2] ADADI, A., AND BERRADA, M. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access* 6 (2018), 52138–52160.
- [3] ADI, Y., KERMANY, E., BELINKOV, Y., LAVI, O., AND GOLDBERG, Y. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv preprint arXiv:1608.04207* (2016).

- [4] ADIWARDANA, D., LUONG, M.-T., SO, D. R., HALL, J., FIEDEL, N., THOPPILAN, R., YANG, Z., KULSHRESHTHA, A., NEMADE, G., LU, Y., ET AL. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977* (2020).
- [5] ALVAREZ-MELIS, D., AND JAAKKOLA, T. S. Towards robust interpretability with self-explaining neural networks. *arXiv preprint arXiv:1806.07538* (2018).
- [6] ANCONA, M., CEOLINI, E., ÖZTIRELI, C., AND GROSS, M. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104* (2017).
- [7] ANTOGNINI, D., AND FALTINGS, B. Rationalization through concepts. *arXiv preprint arXiv:2105.04837* (2021).
- [8] ARKHANGELSKAIA, E., AND DUTTA, S. Whatcha lookin’ at? deeplifting bert’s attention in question answering. *arXiv preprint arXiv:1910.06431* (2019).
- [9] ARRAS, L., HORN, F., MONTAVON, G., MÜLLER, K.-R., AND SAMEK, W. Explaining predictions of non-linear classifiers in nlp. *arXiv preprint arXiv:1606.07298* (2016).
- [10] ARRAS, L., HORN, F., MONTAVON, G., MÜLLER, K.-R., AND SAMEK, W. "what is relevant in a text document?": An interpretable machine learning approach. *PloS one* 12, 8 (2017), e0181142.
- [11] ARRAS, L., MONTAVON, G., MÜLLER, K.-R., AND SAMEK, W. Explaining recurrent neural network predictions in sentiment analysis. *arXiv preprint arXiv:1706.07206* (2017).
- [12] ASAI, A., HASHIMOTO, K., HAJISHIRZI, H., SOCHER, R., AND XIONG, C. Learning to retrieve reasoning paths over wikipedia graph for question answering. *arXiv preprint arXiv:1911.10470* (2019).
- [13] BACH, S., BINDER, A., MONTAVON, G., KLAUSCHEN, F., MÜLLER, K.-R., AND SAMEK, W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* 10, 7 (2015).
- [14] BACH, S., BINDER, A., MONTAVON, G., KLAUSCHEN, F., MÜLLER, K.-R., AND SAMEK, W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE* 10, 7 (07 2015), 1–46.
- [15] BAHDANAU, D., CHO, K., AND BENGIO, Y. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).
- [16] BAHETI, A., RITTER, A., LI, J., AND DOLAN, B. Generating more interesting responses in neural conversation models with distributional constraints. *arXiv preprint arXiv:1809.01215* (2018).
- [17] BANSAL, G., WU, T., ZHOU, J., FOK, R., NUSHI, B., KAMAR, E., RIBEIRO, M. T., AND WELD, D. Does the whole exceed its parts? the effect of ai explanations on complementary team performance. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (2021), pp. 1–16.
- [18] BASHIER, H. K., KIM, M.-Y., AND GOEBEL, R. RANCC: Rationalizing neural networks via concept clustering. In *Proceedings of the 28th International Conference on Computational Linguistics* (Barcelona, Spain (Online), Dec. 2020), International Committee on Computational Linguistics, pp. 3214–3224.
- [19] BASTINGS, J., AZIZ, W., AND TITOV, I. Interpretable neural predictions with differentiable binary variables. *arXiv preprint arXiv:1905.08160* (2019).
- [20] BENGIO, Y., DUCHARME, R., VINCENT, P., AND JANVIN, C. A neural probabilistic language model. *J. Mach. Learn. Res.* 3, null (Mar. 2003), 1137–1155.
- [21] BERTSIMAS, D., DELARUE, A., JAILLET, P., AND MARTIN, S. The price of interpretability. *arXiv preprint arXiv:1907.03419* (2019).
- [22] BODRIA, F., GIANNOTTI, F., GUIDOTTI, R., NARETTO, F., PEDRESCHI, D., AND RINZIVILLO, S. Benchmarking and survey of explanation methods for black box models, 2021.
- [23] BOMMASANI, R., DAVIS, K., AND CARDIE, C. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (Online, July 2020), Association for Computational Linguistics, pp. 4758–4781.
- [24] BOSSELUT, A., RASHKIN, H., SAP, M., MALAVIYA, C., CELIKYILMAZ, A., AND CHOI, Y. Comet: Commonsense transformers for automatic knowledge graph construction. *arXiv preprint arXiv:1906.05317* (2019).

- [25] BOUCHACOURT, D., AND DENOYER, L. Educe: Explaining model decisions through unsupervised concepts extraction. *arXiv preprint arXiv:1905.11852* (2019).
- [26] BOWMAN, S. R., ANGELI, G., POTTS, C., AND MANNING, C. D. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326* (2015).
- [27] BOWMAN, S. R., GAUTHIER, J., RASTOGI, A., GUPTA, R., MANNING, C. D., AND POTTS, C. A fast unified model for parsing and sentence understanding. *arXiv preprint arXiv:1603.06021* (2016).
- [28] BRAŽINSKAS, A., HAVRYLOV, S., AND TITOV, I. Embedding words as distributions with a bayesian skip-gram model. *arXiv preprint arXiv:1711.11027* (2017).
- [29] BROWN, T. B., MANN, B., RYDER, N., SUBBIAH, M., KAPLAN, J., DHARIWAL, P., NEELAKANTAN, A., SHYAM, P., SASTRY, G., ASKELL, A., AGARWAL, S., HERBERT-VOSS, A., KRUEGER, G., HENIGHAN, T., CHILD, R., RAMESH, A., ZIEGLER, D. M., WU, J., WINTER, C., HESSE, C., CHEN, M., SIGLER, E., LITWIN, M., GRAY, S., CHES, B., CLARK, J., BERNER, C., MCCANDLISH, S., RADFORD, A., SUTSKEVER, I., AND AMODEI, D. Language models are few-shot learners.
- [30] BRUNNER, G., LIU, Y., PASCUAL, D., RICHTER, O., CIARAMITA, M., AND WATTENHOFFER, R. On identifiability in transformers. In *International Conference on Learning Representations* (2020).
- [31] CAMBURU, O.-M., ROCKTÄSCHEL, T., LUKASIEWICZ, T., AND BLUNSOM, P. e-snli: Natural language inference with natural language explanations. *arXiv preprint arXiv:1812.01193* (2018).
- [32] CHAI, D., WU, W., HAN, Q., WU, F., AND LI, J. Description based text classification with reinforcement learning, 2020.
- [33] CHANG, S., ZHANG, Y., YU, M., AND JAAKKOLA, T. A game theoretic approach to class-wise selective rationalization. In *Advances in Neural Information Processing Systems* (2019), pp. 10055–10065.
- [34] CHEN, B., FU, Y., XU, G., XIE, P., TAN, C., CHEN, M., AND JING, L. Probing bert in hyperbolic spaces, 2021.
- [35] CHEN, H., AND JI, Y. Learning variational word masks to improve the interpretability of neural text classifiers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Online, Nov. 2020), Association for Computational Linguistics, pp. 4236–4251.
- [36] CHEN, H., ZHENG, G., AND JI, Y. Generating hierarchical explanations on text classification via feature interaction detection. *arXiv preprint arXiv:2004.02015* (2020).
- [37] CHEN, J., SONG, L., WAINWRIGHT, M. J., AND JORDAN, M. I. Learning to explain: An information-theoretic perspective on model interpretation. *arXiv preprint arXiv:1802.07814* (2018).
- [38] CHO, K., VAN MERRIËNBOER, B., GULCEHRE, C., BAHDANAU, D., BOUGARES, F., SCHWENK, H., AND BENGIO, Y. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Doha, Qatar, Oct. 2014), Association for Computational Linguistics, pp. 1724–1734.
- [39] CHOROMANSKI, K., LIKHOSHERSTOV, V., DOHAN, D., SONG, X., GANE, A., SARLÓS, T., HAWKINS, P., DAVIS, J., MOHIUDDIN, A., KAISER, L., BELANGER, D., COLWELL, L., AND WELLER, A. Rethinking attention with performers. *CoRR* (2020).
- [40] CHRYSOSTOMOU, G., AND ALETRAS, N. Variable instance-level explainability for text classification, 2021.
- [41] CLARK, K., KHANDELWAL, U., LEVY, O., AND MANNING, C. D. What does bert look at? an analysis of bert’s attention, 2019.
- [42] CLARK, K., LUONG, M.-T., LE, Q. V., AND MANNING, C. D. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations* (2020).
- [43] CONNEAU, A., KRUSZEWSKI, G., LAMPLE, G., BARRAULT, L., AND BARONI, M. What you can cram into a single $\&\!#\ast$ vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Melbourne, Australia, July 2018), Association for Computational Linguistics, pp. 2126–2136.
- [44] COOK, R. D. Assessment of local influence. *Journal of the Royal Statistical Society: Series B (Methodological)* 48, 2 (1986), 133–155.

- [45] COOK, R. D., AND WEISBERG, S. *Residuals and influence in regression*. New York: Chapman and Hall, 1982.
- [46] CROCE, D., FILICE, S., CASTELLUCCI, G., AND BASILI, R. Deep learning in semantic kernel spaces. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Vancouver, Canada, July 2017), Association for Computational Linguistics, pp. 345–354.
- [47] CROCE, D., ROSSINI, D., AND BASILI, R. Explaining non-linear classifier decisions within kernel-based deep architectures. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP* (Brussels, Belgium, Nov. 2018), Association for Computational Linguistics, pp. 16–24.
- [48] CROCE, D., ROSSINI, D., AND BASILI, R. Auditing deep learning processes through kernel-based explanatory models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (2019), pp. 4028–4037.
- [49] CUTURI, M. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems* 26 (2013), 2292–2300.
- [50] DE CAO, N., SCHLICHTKRULL, M., AZIZ, W., AND TITOV, I. How do decisions emerge across layers in neural models? interpretation with differentiable masking. *arXiv preprint arXiv:2004.14992* (2020).
- [51] DENIL, M., DEMIRAJ, A., AND DE FREITAS, N. Extraction of salient sentences from labelled documents, 2015.
- [52] DEVLIN, J., CHANG, M.-W., LEE, K., AND TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
- [53] DEYOUNG, J., JAIN, S., RAJANI, N. F., LEHMAN, E., XIONG, C., SOCHER, R., AND WALLACE, B. C. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (Online, July 2020), Association for Computational Linguistics.
- [54] DING, Y., LIU, Y., LUAN, H., AND SUN, M. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2017), pp. 1150–1159.
- [55] DONG, L., YANG, N., WANG, W., WEI, F., LIU, X., WANG, Y., GAO, J., ZHOU, M., AND HON, H.-W. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems* (2019), vol. 32, Curran Associates, Inc., pp. 13063–13075.
- [56] DONG, Y., LI, Z., REZAGHOLIZADEH, M., AND CHEUNG, J. C. K. EditNTS: An neural programmer-interpreter model for sentence simplification through explicit editing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (Florence, Italy, July 2019), Association for Computational Linguistics, pp. 3393–3402.
- [57] DROZDOV, A., VERGA, P., YADAV, M., IYYER, M., AND MCCALLUM, A. Unsupervised latent tree induction with deep inside-outside recursive autoencoders. *arXiv preprint arXiv:1904.02142* (2019).
- [58] DUFTER, P., AND SCHÜTZE, H. Analytical methods for interpretable ultradense word embeddings. *arXiv preprint arXiv:1904.08654* (2019).
- [59] EDUNOV, S., OTT, M., AULI, M., AND GRANGIER, D. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381* (2018).
- [60] FENG, S., AND BOYD-GRABER, J. What can ai do for me? evaluating machine learning interpretations in cooperative play. In *Proceedings of the 24th International Conference on Intelligent User Interfaces* (2019), pp. 229–239.
- [61] FENG, S., WALLACE, E., GRISSOM II, A., IYYER, M., RODRIGUEZ, P., AND BOYD-GRABER, J. Pathologies of neural models make interpretations difficult. *arXiv preprint arXiv:1804.07781* (2018).
- [62] GHADER, H., AND MONZ, C. What does attention in neural machine translation pay attention to? In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (Taipei, Taiwan, Nov. 2017), Asian Federation of Natural Language Processing, pp. 30–39.
- [63] GHAEINI, R., FERN, X. Z., AND TADEPALLI, P. Interpreting recurrent and attention-based neural models: a case study on natural language inference. *arXiv preprint arXiv:1808.03894* (2018).
- [64] GHOLIZADEH, S., AND ZHOU, N. Model explainability in deep learning based natural language processing. *arXiv preprint arXiv:2106.07410* (2021).

- [65] GOH, G. S., LAPUSCHKIN, S., WEBER, L., SAMEK, W., AND BINDER, A. Understanding integrated gradients with smoothtaylor for deep neural network attribution. In *2020 25th International Conference on Pattern Recognition (ICPR)* (2021), IEEE, pp. 4949–4956.
- [66] GOLDBERG, Y. Assessing bert’s syntactic abilities. *arXiv preprint arXiv:1901.05287* (2019).
- [67] GREFF, K., SRIVASTAVA, R. K., KOUTNÍK, J., STEUNEBRINK, B. R., AND SCHMIDHUBER, J. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems* 28, 10 (2016), 2222–2232.
- [68] GUAN, C., WANG, X., ZHANG, Q., CHEN, R., HE, D., AND XIE, X. Towards a deep and unified understanding of deep neural models in nlp. In *International conference on machine learning* (2019), PMLR, pp. 2454–2463.
- [69] GUO, H., RAJANI, N. F., HASE, P., BANSAL, M., AND XIONG, C. Fastif: Scalable influence functions for efficient model interpretation and debugging, 2020.
- [70] HAN, X., WALLACE, B. C., AND TSVETKOV, Y. Explaining black box predictions and unveiling data artifacts through influence functions. *arXiv preprint arXiv:2005.06676* (2020).
- [71] HAO, Y., DONG, L., WEI, F., AND XU, K. Visualizing and understanding the effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (Hong Kong, China, Nov. 2019), Association for Computational Linguistics, pp. 4143–4152.
- [72] HASE, P., ZHANG, S., XIE, H., AND BANSAL, M. Leakage-adjusted simulatability: Can models generate non-trivial explanations of their behavior in natural language?, 2020.
- [73] HERMANS, M., AND SCHRAUWEN, B. Training and analysing deep recurrent neural networks. In *Advances in neural information processing systems* (2013), pp. 190–198.
- [74] HEWITT, J., AND LIANG, P. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (Hong Kong, China, Nov. 2019), Association for Computational Linguistics, pp. 2733–2743.
- [75] HEWITT, J., AND MANNING, C. D. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (Minneapolis, Minnesota, June 2019), Association for Computational Linguistics, pp. 4129–4138.
- [76] HINTON, G., VINYALS, O., AND DEAN, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).
- [77] HOCHREITER, S., AND SCHMIDHUBER, J. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
- [78] HTUT, P. M., PHANG, J., BORDIA, S., AND BOWMAN, S. R. Do attention heads in bert track syntactic dependencies?, 2019.
- [79] HUANG, Y., CHEN, Y., DU, Y., AND YANG, Z. Distribution matching for rationalization. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 14 (May 2021), 13090–13097.
- [80] HUPKES, D., VELDHOFEN, S., AND ZUIDEMA, W. Visualisation and ‘diagnostic classifiers’ reveal how recurrent and recursive neural networks process hierarchical structure. *J. Artif. Int. Res.* 61, 1 (Jan. 2018), 907–926.
- [81] HURTADO BODELL, M., ARVIDSSON, M., AND MAGNUSSON, M. Interpretable word embeddings via informative priors. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (Hong Kong, China, Nov. 2019), Association for Computational Linguistics, pp. 6323–6329.
- [82] IRSOY, O., AND CARDIE, C. Deep recursive neural networks for compositionality in language. In *Advances in neural information processing systems* (2014), pp. 2096–2104.
- [83] JACOVI, A., AND GOLDBERG, Y. Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness? *arXiv preprint arXiv:2004.03685* (2020).
- [84] JACOVI, A., AND GOLDBERG, Y. Aligning faithful interpretations with their social attribution. *Transactions of the Association for Computational Linguistics* 9 (2021), 294–310.

- [85] JACOVI, A., SWAYAMDIPTA, S., RAVFOGEL, S., ELAZAR, Y., CHOI, Y., AND GOLDBERG, Y. Contrastive explanations for model interpretability, 2021.
- [86] JAIN, S., AND WALLACE, B. C. Attention is not explanation, 2019.
- [87] JAIN, S., WIEGREFFE, S., PINTER, Y., AND WALLACE, B. C. Learning to faithfully rationalize by construction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (Online, July 2020), Association for Computational Linguistics, pp. 4459–4473.
- [88] JANG, E., GU, S., AND POOLE, B. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144* (2016).
- [89] JAWAHAR, G., SAGOT, B., AND SEDDAH, D. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics* (2019).
- [90] JIANG, Y., AND BANSAL, M. Self-assembling modular networks for interpretable multi-hop reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (Hong Kong, China, Nov. 2019), Association for Computational Linguistics, pp. 4474–4484.
- [91] JIANG, Y., JOSHI, N., CHEN, Y.-C., AND BANSAL, M. Explore, propose, and assemble: An interpretable model for multi-hop reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (Florence, Italy, July 2019), Association for Computational Linguistics, pp. 2714–2725.
- [92] JIN, X., WEI, Z., DU, J., XUE, X., AND REN, X. Towards hierarchical importance attribution: Explaining compositional semantics for neural sequence models. In *International Conference on Learning Representations* (2020).
- [93] JOSHI, M., CHEN, D., LIU, Y., WELD, D. S., ZETTLEMOYER, L., AND LEVY, O. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics* 8 (2020), 64–77.
- [94] KÁDÁR, A., CHRUPAŁA, G., AND ALISHAHI, A. Representation of linguistic form and function in recurrent neural networks. *Computational Linguistics* 43, 4 (2017), 761–780.
- [95] KALCHBRENNER, N., GREFFENSTETTE, E., AND BLUNSOM, P. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188* (2014).
- [96] KARPATHY, A., JOHNSON, J., AND FEI-FEI, L. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078* (2015).
- [97] KARPUKHIN, V., OĞUZ, B., MIN, S., LEWIS, P., WU, L., EDUNOV, S., CHEN, D., AND YIH, W.-T. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906* (2020).
- [98] KIM, B., WATTENBERG, M., GILMER, J., CAI, C., WEXLER, J., VIEGAS, F., ET AL. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning* (2018), PMLR, pp. 2668–2677.
- [99] KIM, S., YI, J., KIM, E., AND YOON, S. Interpretation of NLP models through input marginalization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Online, Nov. 2020), Association for Computational Linguistics, pp. 3154–3167.
- [100] KIM, Y. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).
- [101] KIM, Y., RUSH, A. M., YU, L., KUNCORO, A., DYER, C., AND MELIS, G. Unsupervised recurrent neural network grammars. *arXiv preprint arXiv:1904.03746* (2019).
- [102] KINDERMANS, P.-J., SCHÜTT, K. T., ALBER, M., MÜLLER, K.-R., ERHAN, D., KIM, B., AND DÄHNE, S. Learning how to explain neural networks: Patternnet and patternattribution, 2017.
- [103] KOBAYASHI, S., YOKOI, S., SUZUKI, J., AND INUI, K. Efficient estimation of influence of a training instance. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing* (Online, Nov. 2020), Association for Computational Linguistics, pp. 41–47.
- [104] KOH, P. W., AND LIANG, P. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70* (2017), JMLR. org, pp. 1885–1894.

- [105] KOVALEVA, O., ROMANOV, A., ROGERS, A., AND RUMSHISKY, A. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (Hong Kong, China, Nov. 2019), Association for Computational Linguistics, pp. 4365–4374.
- [106] KOZLOWSKI, A. C., TADDY, M., AND EVANS, J. A. The geometry of culture: Analyzing the meanings of class through word embeddings. *American Sociological Review* 84, 5 (2019), 905–949.
- [107] KUMAR, S., AND TALUKDAR, P. Nile: Natural language inference with faithful natural language explanations. *arXiv preprint arXiv:2005.12116* (2020).
- [108] KUMARASWAMY, P. A generalized probability density function for double-bounded random processes. *Journal of hydrology* 46, 1-2 (1980), 79–88.
- [109] LAI, V., AND TAN, C. On human predictions with explanations and predictions of machine learning models: A case study on deception detection. In *Proceedings of the Conference on Fairness, Accountability, and Transparency* (2019), pp. 29–38.
- [110] LAN, Z., CHEN, M., GOODMAN, S., GIMPEL, K., SHARMA, P., AND SORICUT, R. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942* (2019).
- [111] LAURETIG, A. Identification, interpretability, and Bayesian word embeddings. In *Proceedings of the Third Workshop on Natural Language Processing and Computational Social Science* (Minneapolis, Minnesota, June 2019), Association for Computational Linguistics, pp. 7–17.
- [112] LEE, J., SHIN, J.-H., AND KIM, J.-S. Interactive visualization and manipulation of attention-based neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* (2017), pp. 121–126.
- [113] LEI, T., BARZILAY, R., AND JAAKKOLA, T. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155* (2016).
- [114] LERTVITTAYAKUMJORN, P., AND TONI, F. Explanation-based human debugging of nlp models: A survey, 2021.
- [115] LEWIS, M., LIU, Y., GOYAL, N., GHAZVININEJAD, M., MOHAMED, A., LEVY, O., STOYANOV, V., AND ZETTLEMOYER, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461* (2019).
- [116] LI, J., CHEN, X., HOVY, E., AND JURAFSKY, D. Visualizing and understanding neural models in nlp. *arXiv preprint arXiv:1506.01066* (2015).
- [117] LI, J., LUONG, M.-T., JURAFSKY, D., AND HOVY, E. When are tree structures necessary for deep learning of representations? *arXiv preprint arXiv:1503.00185* (2015).
- [118] LI, J., MONROE, W., AND JURAFSKY, D. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220* (2016).
- [119] LI, O., LIU, H., CHEN, C., AND RUDIN, C. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2018), vol. 32.
- [120] LI, X., FENG, J., MENG, Y., HAN, Q., WU, F., AND LI, J. A unified mrc framework for named entity recognition. *arXiv preprint arXiv:1910.11476* (2019).
- [121] LIN, Y., MENG, Y., SUN, X., HAN, Q., KUANG, K., LI, J., AND WU, F. Bertgen: Transductive text classification by combining gcen and bert. *arXiv preprint arXiv:2105.05727* (2021).
- [122] LIN, Y., TAN, Y. C., AND FRANK, R. Open sesame: Getting inside BERT’s linguistic knowledge. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP* (Florence, Italy, Aug. 2019), Association for Computational Linguistics, pp. 241–253.
- [123] LIN, Z., FENG, M., SANTOS, C. N. D., YU, M., XIANG, B., ZHOU, B., AND BENGIO, Y. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130* (2017).
- [124] LIU, H., YIN, Q., AND WANG, W. Y. Towards explainable nlp: A generative explanation framework for text classification. *arXiv preprint arXiv:1811.00196* (2018).

- [125] LIU, X., HE, P., CHEN, W., AND GAO, J. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (Florence, Italy, July 2019), Association for Computational Linguistics, pp. 4487–4496.
- [126] LUONG, T., PHAM, H., AND MANNING, C. D. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (Lisbon, Portugal, Sept. 2015), Association for Computational Linguistics, pp. 1412–1421.
- [127] MA, X., AND HOVY, E. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354* (2016).
- [128] MAJUMDER, B. P., CAMBURU, O.-M., LUKASIEWICZ, T., AND MCAULEY, J. Rationale-inspired natural language explanations with commonsense, 2021.
- [129] MENG, Y., FAN, C., SUN, Z., HOVY, E., WU, F., AND LI, J. Pair the dots: Jointly examining training history and test stimuli for model interpretability, 2020.
- [130] MIKOLOV, T., SUTSKEVER, I., CHEN, K., CORRADO, G., AND DEAN, J. Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546* (2013).
- [131] MILLER, T. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence* 267 (2019), 1–38.
- [132] MONTAVON, G., LAPUSCHKIN, S., BINDER, A., SAMEK, W., AND MÜLLER, K.-R. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition* 65 (2017), 211–222.
- [133] MURDOCH, W. J., LIU, P. J., AND YU, B. Beyond word importance: Contextual decomposition to extract interactions from lstms. *arXiv preprint arXiv:1801.05453* (2018).
- [134] NARANG, S., RAFFEL, C., LEE, K., ROBERTS, A., FIEDEL, N., AND MALKAN, K. Wt5?! training text-to-text models to explain their predictions. *CoRR* (2020).
- [135] NGUYEN, D. Comparing automatic and human evaluation of local explanations for text classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* (2018), pp. 1069–1078.
- [136] NISHIDA, K., NISHIDA, K., NAGATA, M., OTSUKA, A., SAITO, I., ASANO, H., AND TOMITA, J. Answering while summarizing: Multi-task learning for multi-hop qa with evidence extraction. *arXiv preprint arXiv:1905.08511* (2019).
- [137] PANIGRAHI, A., SIMHADRI, H. V., AND BHATTACHARYYA, C. Word2Sense: Sparse interpretable word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (Florence, Italy, July 2019), Association for Computational Linguistics, pp. 5692–5705.
- [138] PARANJAPPE, B., JOSHI, M., THICKSTUN, J., HAJISHIRZI, H., AND ZETTLEMOYER, L. An information bottleneck approach for controlling conciseness in rationale extraction. *arXiv preprint arXiv:2005.00652* (2020).
- [139] PARK, S., BAK, J., AND OH, A. Rotated word vector representations and their interpretability. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* (2017), pp. 401–411.
- [140] POERNER, N., SCHÜTZE, H., AND ROTH, B. Evaluating neural network explanation methods using hybrid documents and morphosyntactic agreement. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Melbourne, Australia, July 2018), Association for Computational Linguistics, pp. 340–350.
- [141] PRÖLLOCHS, N., FEUERRIEGEL, S., AND NEUMANN, D. Negation scope detection in sentiment analysis: Decision support for news-driven trading. *Decision Support Systems* 88 (2016), 67–75.
- [142] PRÖLLOCHS, N., FEUERRIEGEL, S., AND NEUMANN, D. Learning interpretable negation rules via weak supervision at document level: A reinforcement learning approach. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (Minneapolis, Minnesota, June 2019), Association for Computational Linguistics, pp. 407–413.
- [143] PRUTHI, D., GUPTA, M., DHINGRA, B., NEUBIG, G., AND LIPTON, Z. C. Learning to deceive with attention-based explanations. *arXiv preprint arXiv:1909.07913* (2019).

- [144] RADFORD, A., NARASIMHAN, K., SALIMANS, T., AND SUTSKEVER, I. Improving language understanding by generative pre-training, 2018.
- [145] RADFORD, A., WU, J., CHILD, R., LUAN, D., AMODEI, D., AND SUTSKEVER, I. Language models are unsupervised multitask learners. *OpenAI Blog* 1, 8 (2019).
- [146] RAFFEL, C., SHAZEER, N., ROBERTS, A., LEE, K., NARANG, S., MATENA, M., ZHOU, Y., LI, W., AND LIU, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683* (2019).
- [147] RAJAGOPAL, D., BALACHANDRAN, V., HOVY, E., AND TSVETKOV, Y. Selfexplain: A self-explaining architecture for neural text classifiers, 2021.
- [148] RAJANI, N. F., KRAUSE, B., YIN, W., NIU, T., SOCHER, R., AND XIONG, C. Explaining and improving model behavior with k nearest neighbor representations, 2020.
- [149] RAJANI, N. F., MCCANN, B., XIONG, C., AND SOCHER, R. Explain yourself! leveraging language models for commonsense reasoning. *arXiv preprint arXiv:1906.02361* (2019).
- [150] REIF, E., YUAN, A., WATTENBERG, M. M., VIEGAS, F., COENEN, A., PEARCE, A., AND KIM, B. Visualizing and measuring the geometry of bert.
- [151] REN, M., GENG, X., QIN, T., HUANG, H., AND JIANG, D. Towards interpretable reasoning over paragraph effects in situation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Online, Nov. 2020), Association for Computational Linguistics, pp. 6745–6758.
- [152] RIBEIRO, M. T., SINGH, S., AND GUESTRIN, C. "why should i trust you?": Explaining the predictions of any classifier, 2016.
- [153] RIBEIRO, M. T., SINGH, S., AND GUESTRIN, C. Anchors: High-precision model-agnostic explanations. In *AAAI* (2018), vol. 18, pp. 1527–1535.
- [154] ROGERS, A., KOVALEVA, O., AND RUMSHISKY, A. A primer in bertology: What we know about how bert works, 2020.
- [155] ROSA, R., AND MAREČEK, D. Inducing syntactic trees from bert representations. *arXiv preprint arXiv:1906.11511* (2019).
- [156] ROSS, A. S., HUGHES, M. C., AND DOSHI-VELEZ, F. Right for the right reasons: Training differentiable models by constraining their explanations. *arXiv preprint arXiv:1703.03717* (2017).
- [157] ROTHE, S., AND SCHÜTZE, H. Word embedding calculus in meaningful ultradense subspaces. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (2016), pp. 512–517.
- [158] RUZSICS, T., SOZINOVA, O., GUTIERREZ-VASQUES, X., AND SAMARDZIC, T. Interpretability for morphological inflection: from character-level predictions to subword-level rules. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume* (Online, Apr. 2021), Association for Computational Linguistics, pp. 3189–3201.
- [159] SAP, M., LE BRAS, R., ALLAWAY, E., BHAGAVATULA, C., LOURIE, N., RASHKIN, H., ROOF, B., SMITH, N. A., AND CHOI, Y. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2019), vol. 33, pp. 3027–3035.
- [160] SATO, M., SUZUKI, J., SHINDO, H., AND MATSUMOTO, Y. Interpretable adversarial perturbation in input embedding space for text. *arXiv preprint arXiv:1805.02917* (2018).
- [161] SCHLICHTKRULL, M. S., DE CAO, N., AND TITOV, I. Interpreting graph neural networks for nlp with differentiable edge masking. *arXiv preprint arXiv:2010.00577* (2020).
- [162] SCHWARZENBERG, R., RAITHEL, L., AND HARBECKE, D. Neural vector conceptualization for word vector space interpretation. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP* (Minneapolis, USA, June 2019), Association for Computational Linguistics, pp. 1–7.
- [163] ŞENEL, L. K., UTLU, I., YÜCESOY, V., KOC, A., AND CUKUR, T. Semantic structure and interpretability of word embeddings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 26, 10 (2018), 1769–1779.
- [164] SERRANO, S., AND SMITH, N. A. Is attention interpretable? *arXiv preprint arXiv:1906.03731* (2019).

- [165] SHAO, Y., HARDMEIER, C., TIEDEMANN, J., AND NIVRE, J. Character-based joint segmentation and pos tagging for chinese using bidirectional rnn-crf. *arXiv preprint arXiv:1704.01314* (2017).
- [166] SHEN, Y., LIN, Z., HUANG, C.-W., AND COURVILLE, A. Neural language modeling by jointly learning syntax and lexicon. *arXiv preprint arXiv:1711.02013* (2017).
- [167] SHEN, Y., TAN, S., HOSSEINI, A., LIN, Z., SORDONI, A., AND COURVILLE, A. Ordered memory. *arXiv preprint arXiv:1910.13466* (2019).
- [168] SHEN, Y., TAN, S., SORDONI, A., AND COURVILLE, A. Ordered neurons: Integrating tree structures into recurrent neural networks. *arXiv preprint arXiv:1810.09536* (2018).
- [169] SHI, X., PADHI, I., AND KNIGHT, K. Does string-based neural mt learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (2016), pp. 1526–1534.
- [170] SHIN, J., MADOTTO, A., AND FUNG, P. Interpreting word embeddings with eigenvector analysis. In *32nd Conference on Neural Information Processing Systems (NIPS 2018), IRASL workshop* (2018).
- [171] SHRIKUMAR, A., GREENSIDE, P., AND KUNDAJE, A. Learning important features through propagating activation differences. *arXiv preprint arXiv:1704.02685* (2017).
- [172] SIMONYAN, K., VEDALDI, A., AND ZISSERMAN, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* (2013).
- [173] SINGH, C., MURDOCH, W. J., AND YU, B. Hierarchical interpretations for neural network predictions. *arXiv preprint arXiv:1806.05337* (2018).
- [174] SMILKOV, D., THORAT, N., KIM, B., VIÉGAS, F., AND WATTENBERG, M. Smoothgrad: removing noise by adding noise, 2017.
- [175] SOCHER, R., PERELYGIN, A., WU, J., CHUANG, J., MANNING, C. D., NG, A., AND POTTS, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing* (2013), pp. 1631–1642.
- [176] SOCHER, R., PERELYGIN, A., WU, J., CHUANG, J., MANNING, C. D., NG, A., AND POTTS, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing* (Seattle, Washington, USA, Oct. 2013), Association for Computational Linguistics, pp. 1631–1642.
- [177] SONG, K., TAN, X., QIN, T., LU, J., AND LIU, T.-Y. Mass: Masked sequence to sequence pre-training for language generation. In *International Conference on Machine Learning* (2019), pp. 5926–5936.
- [178] SRINIVAS, S., AND FLEURET, F. Full-gradient representation for neural network visualization. In *Advances in Neural Information Processing Systems* (2019), pp. 4124–4133.
- [179] STROBELT, H., GEHRMANN, S., HUBER, B., PFISTER, H., RUSH, A. M., ET AL. Visual analysis of hidden state dynamics in recurrent neural networks. *arXiv preprint arXiv:1606.07461* (2016).
- [180] SUBRAMANIAN, A., PRUTHI, D., JHAMTANI, H., BERG-KIRKPATRICK, T., AND HOVY, E. Spine: Sparse interpretable neural embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence* (2018).
- [181] SUN, Y., WANG, S., LI, Y., FENG, S., CHEN, X., ZHANG, H., TIAN, X., ZHU, D., TIAN, H., AND WU, H. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223* (2019).
- [182] SUN, Z., FAN, C., HAN, Q., SUN, X., MENG, Y., WU, F., AND LI, J. Self-explaining structures improve nlp models, 2020.
- [183] SUN, Z., LI, X., SUN, X., MENG, Y., AO, X., HE, Q., WU, F., AND LI, J. Chinesebert: Chinese pretraining enhanced by glyph and pinyin information. *arXiv preprint arXiv:2106.16038* (2021).
- [184] SUNDARARAJAN, M., TALY, A., AND YAN, Q. Axiomatic attribution for deep networks. *arXiv preprint arXiv:1703.01365* (2017).
- [185] SWANSON, K., YU, L., AND LEI, T. Rationalizing text matching: Learning sparse alignments via optimal transport. *arXiv preprint arXiv:2005.13111* (2020).

- [186] TAI, K. S., SOCHER, R., AND MANNING, C. D. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075* (2015).
- [187] TAKASE, S., AND KIYONO, S. Lessons on parameter sharing across layers in transformers. *arXiv preprint arXiv:2104.06022* (2021).
- [188] TALMOR, A., ELAZAR, Y., GOLDBERG, Y., AND BERANT, J. olympics—on what language model pre-training captures. *arXiv preprint arXiv:1912.13283* (2019).
- [189] TALMOR, A., HERZIG, J., LOURIE, N., AND BERANT, J. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (Minneapolis, Minnesota, June 2019), Association for Computational Linguistics, pp. 4149–4158.
- [190] TENNEY, I., DAS, D., AND PAVLICK, E. Bert rediscovered the classical nlp pipeline. *arXiv preprint arXiv:1905.05950* (2019).
- [191] TJOA, E., AND GUAN, C. A survey on explainable artificial intelligence (xai): Toward medical xai. *IEEE Transactions on Neural Networks and Learning Systems* (2020).
- [192] TONEVA, M., AND WEHBE, L. Interpreting and improving natural-language processing (in machines) with natural language-processing (in the brain). *arXiv preprint arXiv:1905.11833* (2019).
- [193] VASHISHTH, S., UPADHYAY, S., TOMAR, G. S., AND FARUQUI, M. Attention interpretability across nlp tasks. *arXiv preprint arXiv:1909.11218* (2019).
- [194] VASWANI, A., SHAZEER, N., PARMAR, N., USZKOREIT, J., JONES, L., GOMEZ, A. N., KAISER, L. U., AND POLOSUKHIN, I. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5998–6008.
- [195] VIG, J., AND BELINKOV, Y. Analyzing the structure of attention in a transformer language model. *arXiv preprint arXiv:1906.04284* (2019).
- [196] VOITA, E., TALBOT, D., MOISEEV, F., SENNRICH, R., AND TITOV, I. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (Florence, Italy, July 2019), Association for Computational Linguistics, pp. 5797–5808.
- [197] WALLACE, E., FENG, S., AND BOYD-GRABER, J. Interpreting neural networks with nearest neighbors. *arXiv preprint arXiv:1809.02847* (2018).
- [198] WALLACE, E., TUYLS, J., WANG, J., SUBRAMANIAN, S., GARDNER, M., AND SINGH, S. Allennlp interpret: A framework for explaining predictions of nlp models. *arXiv preprint arXiv:1909.09251* (2019).
- [199] WANG, G., LI, C., WANG, W., ZHANG, Y., SHEN, D., ZHANG, X., HENAO, R., AND CARIN, L. Joint embedding of words and labels for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2018), pp. 2321–2331.
- [200] WANG, S., FANG, H., KHABSA, M., MAO, H., AND MA, H. Entailment as few-shot learner. *arXiv preprint arXiv:2104.14690* (2021).
- [201] WANG, Y., HUANG, M., ZHU, X., AND ZHAO, L. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing* (2016), pp. 606–615.
- [202] WIEGREFFE, S., MARASOVIC, A., AND SMITH, N. A. Measuring association between labels and free-text rationales, 2020.
- [203] WIEGREFFE, S., AND PINTER, Y. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (Hong Kong, China, Nov. 2019), Association for Computational Linguistics, pp. 11–20.
- [204] WILLIAMS, C., AND SEEGER, M. Using the nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems* (2001), T. Leen, T. Dietterich, and V. Tresp, Eds., vol. 13, MIT Press.
- [205] WILLIAMS, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8 (1992), 229–256.

- [206] WU, Z., CHEN, Y., KAO, B., AND LIU, Q. Perturbed masking: Parameter-free probing for analyzing and interpreting bert. *arXiv preprint arXiv:2004.14786* (2020).
- [207] XU, J., REN, X., LIN, J., AND SUN, X. Dp-gan: diversity-promoting generative adversarial network for generating informative and diversified text. *arXiv preprint arXiv:1802.01345* (2018).
- [208] YAMADA, I., ASAI, A., SHINDO, H., TAKEDA, H., AND MATSUMOTO, Y. Luke: deep contextualized entity representations with entity-aware self-attention. *arXiv preprint arXiv:2010.01057* (2020).
- [209] YANG, Y., MALAVIYA, C., FERNANDEZ, J., SWAYAMDIPTA, S., BRAS, R. L., WANG, J.-P., BHAGAVATULA, C., CHOI, Y., AND DOWNEY, D. Generative data augmentation for commonsense reasoning, 2020.
- [210] YANG, Z., DAI, Z., YANG, Y., CARBONELL, J., SALAKHUTDINOV, R. R., AND LE, Q. V. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems* (2019), pp. 5753–5763.
- [211] YOSINSKI, J., CLUNE, J., NGUYEN, A., FUCHS, T., AND LIPSON, H. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579* (2015).
- [212] YU, M., CHANG, S., ZHANG, Y., AND JAAKKOLA, T. Rethinking cooperative rationalization: Introspective extraction and complement control. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (Hong Kong, China, Nov. 2019), Association for Computational Linguistics, pp. 4094–4103.
- [213] ZELLINGER, W., GRUBINGER, T., LUGHOFFER, E., NATSCHLÄGER, T., AND SAMINGER-PLATZ, S. Central moment discrepancy (cmd) for domain-invariant representation learning, 2019.
- [214] ZHANG, X., ZHAO, J., AND LECUN, Y. Character-level convolutional networks for text classification. *arXiv preprint arXiv:1509.01626* (2015).
- [215] ZHANG, Y., LIAO, Q. V., AND BELLAMY, R. K. Effect of confidence and explanation on accuracy and trust calibration in ai-assisted decision making. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency* (2020), pp. 295–305.
- [216] ZHANG, Y., TIÑO, P., LEONARDIS, A., AND TANG, K. A survey on neural network interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence* (2021).
- [217] ZHANG, Y., AND YANG, J. Chinese ner using lattice lstm. *arXiv preprint arXiv:1805.02023* (2018).
- [218] ZHANG, Z., YANG, J., AND ZHAO, H. Retrospective reader for machine reading comprehension. *arXiv preprint arXiv:2001.09694* (2020).
- [219] ZHAO, J., ZHOU, Y., LI, Z., WANG, W., AND CHANG, K.-W. Learning gender-neutral word embeddings. *arXiv preprint arXiv:1809.01496* (2018).
- [220] ZHAO, Y., AND BETHARD, S. How does BERT’s attention change when you fine-tune? an analysis methodology and a case study in negation scope. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (Online, July 2020), Association for Computational Linguistics, pp. 4729–4747.
- [221] ZHOU, C., SUN, C., LIU, Z., AND LAU, F. A c-lstm neural network for text classification. *arXiv preprint arXiv:1511.08630* (2015).
- [222] ZHU, J., XIA, Y., WU, L., HE, D., QIN, T., ZHOU, W., LI, H., AND LIU, T.-Y. Incorporating bert into neural machine translation. *arXiv preprint arXiv:2002.06823* (2020).