Interpreting Neural Networks With Nearest Neighbors

Eric Wallace*, **Shi Feng***, **Jordan Boyd-Graber**
University of Maryland
{ewallac2, shifeng, jbg}@umiacs.umd.edu

Abstract

Local model interpretation methods explain individual predictions by assigning an importance value to each input feature. This value is often determined by measuring the change in confidence when a feature is removed. However, the confidence of neural networks is not a robust measure of model uncertainty. This issue makes reliably judging the importance of the input features difficult. We address this by changing the test-time behavior of neural networks using Deep k-Nearest Neighbors. Without harming text classification accuracy, this algorithm provides a more robust uncertainty metric which we use to generate feature importance values. The resulting interpretations better align with human perception than baseline methods. Finally, we use our interpretation method to analyze model predictions on dataset annotation artifacts.

1 Introduction

The growing use of neural networks in sensitive domains such as medicine, finance, and security raises concerns about human trust in these machine learning systems. A central question is test-time *interpretability*: how can humans understand the reasoning behind model predictions?

A common way to interpret neural network predictions is to identify the most important input features. For instance, a saliency map that highlights important pixels in an image (Sundararajan et al., 2017) or words in a sentence (Li et al., 2016). Given a test prediction, the *importance* of each input feature is the change in model confidence when that feature is removed.

However, neural network confidence is not a proper measure of model uncertainty (Guo et al., 2017). This issue is emphasized when models make highly confident predictions on inputs that are completely void of information, for example, images of pure noise (Goodfellow et al., 2015) or meaningless text snippets (Feng et al., 2018). Consequently, a model's confidence may not properly reflect whether discriminative input features are present. This issue makes it difficult to reliably judge the importance of each input feature using common confidence-based interpretation methods (Feng et al., 2018).

To address this, we apply Deep k-Nearest Neighbors (DKNN) (Papernot and McDaniel, 2018) to neural models for text classification. Concretely, predictions are no longer made with a softmax classifier, but using the labels of the *training examples* whose representations are most similar to the test example (Section 3). This provides an alternative metric for model uncertainty, *conformity*, which measures how much support a test prediction has by comparing its hidden representations to the training data. This representation-based uncertainty measurement can be used in combination with existing interpretation methods, such as leave-one-out (Li et al., 2016), to better identify important input features.

We combine DKNN with CNN and LSTM models on six NLP text classification tasks, including sentiment analysis and textual entailment, with no loss in classification accuracy (Section 4). We compare interpretations generated using DKNN conformity to baseline interpretation methods, finding DKNN interpretations rarely assign importance to extraneous words that do not align with human perception (Section 5). Finally, we generate interpretations using DKNN conformity for a dataset with known artifacts (SNLI), helping to indicate whether a model has learned superficial patterns. We open source the code for DKNN and our results.[1]

*Equal contribution

[1] https://github.com/Eric-Wallace/deep-knn

2 Interpretation Through Feature Attribution

Feature attribution methods explain a test prediction by assigning an importance value to each input feature (typically pixels or words).

In the case of text classification, we have an input sequence of n words $\mathbf{x} = \langle w_1, w_2, \dots w_n \rangle$, represented as one-hot vectors. The word sequence is then converted to a sequence of word embeddings $\mathbf{e} = \langle \boldsymbol{v}_1, \boldsymbol{v}_2, \dots \boldsymbol{v}_n \rangle$. A classifier f outputs a probability distribution over classes. The class with the highest probability is selected as the prediction y, with its probability serving as the model confidence. To create an interpretation, each input word is assigned an importance value, $g(w_i \mid \mathbf{x}, y)$, which indicates the word's contribution to the prediction. A saliency map (or heat map) visually highlights the words in a sentence.

2.1 Leave-one-out Attribution

A simple way to define the importance g is via leave-one-out (Li et al., 2016): individually remove a word from the input and see how the confidence changes. The importance of word w_i is the decrease in confidence[2] when word i is removed:

$$g(w_i \mid \mathbf{x}, y) = f(y \mid \mathbf{x}) - f(y \mid \mathbf{x}_{-i}), \quad (1)$$

where $\mathbf{x}_{-i}$ is the input sequence with the ith word removed and $f(y \mid \mathbf{x})$ is the model confidence for class y. This can be repeated for all words in the input. Under this definition, the sign of the importance value is opposite the sign of the confidence change: if a word's removal causes a decrease in the confidence, it gets a positive importance value. We refer to this interpretation method as *Confidence* leave-one-out in our experiments.

2.2 Gradient-Based Feature Attribution

In the case of neural networks, the model $f(\mathbf{x})$ as a function of word w_i is a highly non-linear, differentiable function. Rather than leaving one word out at a time, we can simulate a word's removal by approximating f with a function that is linear in w_i through the first-order Taylor expansion. The importance of w_i is computed as the derivative of f with respect to the one-hot vector:

$$\frac{\partial f}{\partial w_i} = \frac{\partial f}{\partial \boldsymbol{v}_i} \frac{\partial \boldsymbol{v}_i}{\partial w_i} = \frac{\partial f}{\partial \boldsymbol{v}_i} \cdot \boldsymbol{v}_i \quad (2)$$

[2] equivalently the change in class score or cross entropy loss

Thus, a word's importance is the dot product between the gradient of the class prediction with respect to the embedding and the word embedding itself. This gradient approximation simulates the change in confidence when an input word is removed and has been used in various interpretation methods for NLP (Arras et al., 2016; Ebrahimi et al., 2017). We refer to this interpretation approach as *Gradient* in our experiments.

2.3 Interpretation Method Failures

Interpreting neural networks can have unexpected negative results. Ghorbani et al. (2017) and Kindermans et al. (2017) show how a lack of model robustness and stability can cause egregious interpretation failures in computer vision settings. Feng et al. (2018) extend this to NLP and draw connections between interpretation failures and adversarial examples (Szegedy et al., 2014). To counteract this, new interpretation methods alone are not enough—models must be improved. For instance, Feng et al. (2018) argue that interpretation methods should not rely on prediction confidence as it does not reflect a model's uncertainty.

Following this, we improve interpretations by replacing the softmax confidence with a more robust uncertainty estimate using DKNN (Papernot and McDaniel, 2018). This algorithm maintains the accuracy of standard image classification models while providing a better uncertainty metric capable of defending against adversarial examples.

3 Deep k-Nearest Neighbors for Sequential Inputs

This section describes Deep k-Nearest Neighbors, its application to sequential inputs, and how we use it to determine word importance values.

3.1 Deep k-Nearest Neighbors

Papernot and McDaniel (2018) propose Deep k-Nearest Neighbors (DKNN), a modification to the test-time behavior of neural networks.

After training completes, the DKNN algorithm passes every training example through the model and saves each of the layer's representations. This creates a new dataset, whose features are the representations and whose labels are the model predictions. Test-time predictions are made by passing an example through the model and performing k-nearest neighbors classification on the resulting representations. This modification does not de-

grade the accuracy of image classifiers on several standard datasets (Papernot and McDaniel, 2018).

For our purposes, the benefit of DKNN is the algorithm's uncertainty metric, the *conformity* score. This score is the percentage of nearest neighbors belonging to the predicted class. Conformity follows from the framework of conformal prediction (Shafer and Vovk, 2008) and estimates how much the training data supports a classification decision.

The conformity score uses the representations at each neural network layer, and therefore, a prediction only receives high conformity if it largely agrees with the training data at all representation levels. This mechanism defends against adversarial examples (Szegedy et al., 2014), as it is difficult to construct a perturbation which changes the neighbors at every layer. Consequently, conformity is a better uncertainty metric for both regular examples and out-of-domain examples such as noisy or adversarial inputs, making it suitable for interpreting models.

3.2 Handling Sequences

The DKNN algorithm requires fixed-size vector representations. To reach a fixed-size representation for text classification, we take either the final hidden state of a recurrent neural network or use max pooling across time (Collobert and Weston, 2008). We consider deep architectures of these two forms, using each of the layers' representations as the features for DKNN.

3.3 Conformity leave-one-out

Using conformity, we generate interpretations through a modified version of leave-one-out (Li et al., 2016). After removing a word, rather than observing the drop in confidence, we instead measure the drop in conformity. Formally, we modify classifier f in Equation 1 to output probabilities based on conformity. We refer to this method as *conformity leave-one-out*.

4 DKNN Maintains Classification Accuracy

Interpretability should not come at the cost of performance—before investigating how interpretable DKNN is, we first evaluate its accuracy. We experiment with six text classification tasks and two models, verifying that DKNN achieves accuracy comparable to regular classifiers.

4.1 Datasets and Models

We consider six common text classification tasks: binary sentiment analysis using Stanford Sentiment Treebank (Socher et al., 2013, SST) and Customer Reviews (Hu and Liu, 2004, CR), topic classification using *TREC* (Li and Roth, 2002), opinion polarity (Wiebe et al., 2005, MPQA), and subjectivity/objectivity (Pang and Lee, 2004, SUBJ). Additionally, we consider natural language inference with SNLI (Bowman et al., 2015). We experiment with BILSTM and CNN models.

CNN Our CNN architecture resembles Kim (2014). We use convolutional filters of size three, four, and five, with max-pooling over time (Collobert and Weston, 2008). The filters are followed by three fully-connected layers. We fine-tune GLOVE embeddings (Pennington et al., 2014) of each word. For DKNN, we use the activations from the convolution layer and the three fully-connected layers.

BILSTM Our architecture uses a bidirectional LSTM (Graves and Schmidhuber, 2005), with the final hidden state forming the fixed-size representation. We use three LSTM layers, followed by two fully-connected layers. We fine-tune GLOVE embeddings of each word. For DKNN, we use the final activations of the three recurrent layers and the two fully-connected layers.

SNLI Classifier Unlike the other tasks which consist of a single input sentence, SNLI has two inputs, a premise and hypothesis. Following Conneau et al. (2017), we use the same model to encode the two inputs, generating representations u for the premise and v for the hypothesis. We concatenate these two representations along with their dot-product and element-wise absolute difference, arriving at a final representation $[u; v; u * v; |u - v|]$. This vector passes through two fully-connected layers for classification. For DKNN, we use the activations of the two fully-connected layers.

Nearest Neighbor Search For accurate interpretations, we trade efficiency for accuracy and replace locally sensitive hashing (Gionis et al., 1999) used by Papernot and McDaniel (2018) with a k-d tree (Bentley, 1975). We use $k = 75$ nearest neighbors at each layer. The empirical results are robust to the choice of k.

4.2 Classification Results

DKNN achieves comparable accuracy on the five classification tasks (Table 1). On SNLI, the BILSTM achieves an accuracy of 81.2% with a softmax classifier and 81.0% with DKNN.

5 DKNN is Interpretable

Following past work (Li et al., 2016; Murdoch et al., 2018), we focus on the SST dataset for generating interpretations. Due to the lack of standard interpretation evaluation metrics (Doshi-Velez and Kim, 2017), we use qualitative evaluations (Smilkov et al., 2017; Sundararajan et al., 2017; Li et al., 2016), performing quantitative experiments where possible to examine the distinction between the interpretation methods.

5.1 Interpretation Analysis

We compare our method (*Conformity leave-one-out*) against two baselines: leave-one-out using regular confidence (*Confidence leave-one-out*, see Section 2.1) and the gradient with respect to the input (*Gradient*, see Section 2.2). To create saliency maps, we normalize each word's importance by dividing it by the total importance of the words in the sentence. We display unknown words in angle brackets <>. Table 2 shows SST interpretation examples for the BILSTM model and further examples are shown on a supplementary website.[3]

Conformity leave-one-out assigns concentrated importance values to a small number of input words. In contrast, the baseline methods assign non-zero importance values to numerous words, many of which are irrelevant. For instance, in all three examples of Table 2, both baselines highlight almost half of the input, including words such as "fiction" and "clash". We suspect model confidence is oversensitive to these unimportant input changes, causing the baseline interpretations to highlight unimportant words. On the other hand, the conformity score better separates word importance, generating clearer interpretations.

The tendency for confidence-based approaches to assign importance to many words holds for the entire test set. We compute the average number of highlighted words using a threshold of 0.05 (a normalized importance value corresponding to a light blue or light red highlight). Out of the average 20.23 words in SST test set, gradient highlights 5.32 words, confidence leave-one-out highlights 5.79 words, and conformity leave-one-out highlights 3.65 words.

The second, and related, observation for confidence-based approaches is a bias towards selecting word importance based on a word's inherent sentiment, rather than its meaning in context. For example, see "clash", "terribly", and "unfaithful" in Table 2. The removal of these words causes a small change in the model confidence. When using DKNN, the conformity score indicates that the model's uncertainty has not risen without these input words and leave-one-out does not assign them any importance.

We characterize our interpretation method as significantly higher precision, but slightly lower recall than confidence-based methods. Conformity leave-one-out rarely assigns high importance to words that do not align with human perception of sentiment. However, there are cases when our method does not assign significant importance to any word. This occurs when the input has a high redundancy. For example, a positive movie review that describes the sentiment in four distinct ways. In these cases, leaving out a single sentiment word has little effect on the conformity as the model's representation remains supported by the other redundant features. Confidence-based interpretations, which interpret models using the linear units that produce class scores, achieve higher recall by responding to every change in the input for a certain direction but may have lower precision.

In the second example of Table 2, the word "terribly" is assigned a negative importance value, disregarding its positive meaning in context. To examine if this is a stand-alone example or a more general pattern of uninterpretable behavior, we calculate the importance value of the word "terribly" in other positive examples. For each occurrence of the word "great" in positive validation examples, we paraphrase it to "awesome", "wonderful", or "impressive", and add the word "terribly" in front of it. This process yields 66 examples. For each of these examples, we compute the importance value of each input word and rank them from most negative to most positive (the most negative word has a rank of 1). We compare the average ranking of "terribly" from the three methods: 7.9 from conformity leave-one-out, 1.68 from confidence leave-one-out, and 1.1 from gradient. The baseline methods consistently rank "terribly"

[3] https://sites.google.com/view/language-dknn/

	SST	CR	TREC	MPQA	SUBJ
LSTM	86.7	82.7	91.5	88.9	94.8
LSTM DKNN	86.6	82.5	91.3	88.6	94.9
CNN	85.7	83.3	92.8	89.1	93.5
CNN DKNN	85.8	83.4	92.4	88.7	93.1

Table 1: Replacing a neural network's softmax classifier with DKNN maintains classification accuracy on standard text classification tasks.

Method	**Saliency Map**
Conformity	an intelligent fiction about learning through cultural clash.
Confidence	an intelligent fiction about learning through cultural clash.
Gradient	an intelligent fiction about learning through cultural clash.
Conformity	<Schweiger> is talented and terribly charismatic.
Confidence	<Schweiger> is talented and terribly charismatic.
Gradient	<Schweiger> is talented and terribly charismatic.
Conformity	Diane Lane shines in unfaithful.
Confidence	Diane Lane shines in unfaithful.
Gradient	Diane Lane shines in unfaithful.
Color Legend	Positive Impact Negative Impact

Table 2: Comparison of interpretation approaches on SST test examples for the LSTM model. Blue indicates positive impact and red indicates negative impact. Our method (*Conformity* leave-one-out) has higher precision, rarely assigning importance to extraneous words such as "about" or "movie".

as the most negative word, ignoring its meaning in context. This echoes our suspicion: DKNN generates interpretations with higher precision because conformity is robust to irrelevant input changes.

5.2 Analyzing Dataset Annotation Artifacts

We use conformity leave-one-out to interpret a model trained on SNLI, a dataset known to contain annotation artifacts. We demonstrate that our interpretation method can help identify when models exploit dataset biases.

Recent studies (Gururangan et al., 2018; Poliak et al., 2018) identify annotation artifacts in SNLI. Superficial patterns exist in the input which strongly correlate with certain labels, making it possible for models to "game" the task: obtain high accuracy without true understanding. For instance, the hypothesis of an entailment example is often a general paraphrase of the premise, using words such as "outside" instead of "playing in a park". Contradiction examples often contain negation words or non-action verbs like "sleeping". Models trained solely on the hypothesis can learn these patterns and reach accuracies considerably higher than the majority baseline.

These studies indicate that the SNLI task can be gamed. We look to confirm that some artifacts are indeed exploited by normally trained models that use full input pairs. We create saliency maps for examples in the validation set using conformity leave-one-out. Table 3 shows samples and more can be found on the supplementary website. We use blue highlights to indicate words which positively support the model's predicted class, and the color red to indicate words that support a different class. The first example is a randomly sampled baseline, showing how the words "swims" and "pool" support the model's prediction of contradiction. The other examples are selected because they contain terms identified as artifacts. In the second example, conformity leave-one-out assigns extremely high word importance to "sleeping", disregarding the other words necessary to predict contradiction (i.e., the neutral class is still possible if "pets" is replaced with "people"). In the final two hypotheses, the interpretation method

diagnoses the model failure, assigning high importance to "wearing", rather than focusing positively on the shirt color.

To explore this further, we analyze the hypotheses in each SNLI class which contain a top five artifact identified by Gururangan et al. (2018). For each of these examples, we compute the importance value for each input word using both confidence and conformity leave-one-out. We then rank the words from most important for the prediction to least important (a score of 1 indicates highest importance) and report the average rank for the artifacts in Table 4. We sort the words by their Pointwise Mutual Information with the correct label as provided by Gururangan et al. (2018). The word "nobody" particularly stands out: it is the most important input word every time it appears in a contradiction example.

For most of the artifacts, conformity leave-one-out assigns them a high importance, often ranking the artifacts as the most important input word. Confidence leave-one-out correlates less strongly with the known artifacts, frequently ranking them as low as the fifth or sixth most important word. Given the high correlation between conformity leave-one-out and the manually identified artifacts, this interpretation method may serve as a technique to identify undesirable biases a model has learned.

6 Discussion and Related Work

We connect the improvements made by conformity leave-one-out to model confidence issues, compare alternative interpretation improvements, and discuss further features of DKNN.

6.1 Issues in Neural Network Confidence

Many existing feature attribution methods rely on estimates of model uncertainty: both input gradient and confidence leave-one-out rely on prediction confidence, our method relies on DKNN conformity. Interpretation quality is thus determined by reliable uncertainty estimation. For instance, past work shows relying on neural network confidence can lead to unreasonable interpretations (Kindermans et al., 2017; Ghorbani et al., 2017; Feng et al., 2018). Independent of interpretability, Guo et al. (2017) show that neural network confidence is unreasonably high: on held-out examples, it far exceeds empirical accuracy. This is further exemplified by the high confidence predictions produced on inputs that are adversarial (Szegedy et al., 2014) or contain solely noise (Goodfellow et al., 2015).

6.2 Confidence Calibration is Insufficient

We attribute one interpretation failure to neural network confidence issues. Guo et al. (2017) study overconfidence and propose a calibration procedure using Platt scaling, which adjusts the temperature parameter of the softmax function to align confidence with accuracy on a held-out dataset. However, this is not input dependent—the confidence is lower for both full-length examples and ones with words left out. Hence, selecting influential words will remain difficult.

To verify this, we create an interpretation baseline using temperature scaling. The results corroborate the intuition: calibrating the confidence of leave-one-out does not improve interpretations. Qualitatively, the calibrated interpretation results remain comparable to confidence leave-one-out. Furthermore, calibrating the DKNN conformity score as in Papernot and McDaniel (2018) did not improve interpretability compared to the uncalibrated conformity score.

6.3 Alternative Interpretation Improvements

Recent work improves interpretation methods through other means. Smilkov et al. (2017) and Sundararajan et al. (2017) both aggregate gradient values over multiple backpropagation passes to eliminate local noise or satisfy interpretation axioms. This work does not address model confidence and is orthogonal to our DKNN approach.

6.4 Interpretation Through Data Selection

Retrieval-Augmented Convolutional Neural Networks (Zhao and Cho, 2018) are similar to DKNN: they augment model predictions with an information retrieval system that searches over network activations from the training data.

Retrieval-Augmented models and DKNN can both select influential training examples for a test prediction. In particular, the training data activations which are closest to the test point's activations are influential according to the model. These training examples can provide interpretations as a form of analogy (Caruana et al., 1999), an intuitive explanation for both machine learning experts and non-experts (Klein, 1989; Kim et al., 2014; Koh and Liang, 2017; Wallace and Boyd-Graber, 2018). However, unlike in computer vi-

Prediction	Input	Saliency Map
Contradiction	Premise	a young boy reaches for and touches the propeller of a vintage aircraft.
	Hypothesis	a young boy swims in his pool.
Entailment	Premise	a brown a dog and a black dog in the edge of the ocean with a wave under them boats are on the water in the background.
	Hypothesis	the pets are sleeping on the grass..
	Premise	man in a blue shirt standing in front of a structure painted with geometric designs.
Entailment	Hypothesis	a man is wearing a blue shirt.
Entailment	Hypothesis	a man is wearing a black shirt.

Color Legend Positive Impact Negative Impact

Table 3: Interpretations generated with conformity leave-one-out align with annotation biases identified in SNLI. In the second example, the model puts emphasis on the word "sleeping", disregarding other words that could indicate the Neutral class. The final example diagnoses a model's incorrect Entailment prediction (shown in red). Green highlights indicate words that support the classification decision made (shown in parenthesis), pink highlights indicate words that support a different class.

Label	Artifact	Conformity	Confidence
Entailment	outdoors	2.93	3.26
	least	2.22	4.41
	instrument	3.57	4.47
	outside	4.08	4.80
	animal	2.00	4.73
Neutral	tall	1.09	2.61
	first	2.14	2.99
	competition	2.33	5.56
	sad	1.39	1.79
	favorite	1.69	3.89
Contradiction	nobody	1.00	1.00
	sleeping	1.64	2.34
	no	2.53	5.74
	tv	1.92	3.74
	cat	1.42	3.62

Table 4: The top SNLI artifacts identified by Gururangan et al. (2018) are shown on the left. For each word, we compute the average importance rank over the validation set using either *Conformity* or *Confidence* leave-one-out. A score of 1 indicates that a word is always ranked as the most important word in the input. *Conformity* leave-one-out assigns higher importance to artifacts, suggesting it better diagnoses model biases.

sion where training data selection using DKNN yielded interpretable examples (Papernot and McDaniel, 2018), our experiments did not find human interpretable data points for SST or SNLI.

6.5 Trust in Model Predictions

Model confidence is important for real-world applications: it signals how much one should trust a neural network's predictions. Unfortunately, users may be misled when a model outputs highly confident predictions on rubbish examples (Goodfellow et al., 2015; Nguyen et al., 2015) or adversarial examples (Szegedy et al., 2014). Recent work decides when to trust a neural network model (Ribeiro et al., 2016; Doshi-Velez and Kim, 2017; Jiang et al., 2018). For instance, analyzing local linear model approximations (Ribeiro et al., 2016) or flagging rare network activations using kernel density estimation (Jiang et al., 2018). The DKNN conformity score is a trust metric that helps defend against image adversarial examples (Papernot and McDaniel, 2018). Future work should study if this robustness extends to interpretations.

7 Future Work and Conclusion

A robust estimate of model uncertainty is critical to determine feature importance. The DKNN conformity score is one such uncertainty metric which leads to higher precision interpretations. Al-

though DKNN is only a test-time improvement—the model is still trained using maximum likelihood. Combining nearest neighbor and maximum likelihood objectives during training may further improve model accuracy and interpretability. Moreover, other uncertainty estimators do not require test-time modifications. For example, modeling $p(x)$ and $p(y \mid \mathbf{x})$ using Bayesian Neural Networks (Gal et al., 2016).

Similar to other NLP interpretation methods (Sundararajan et al., 2017; Li et al., 2016), conformity leave-one-out works when a model's representation has a fixed size. For other NLP tasks, such as structured prediction (e.g., translation and parsing) or span prediction (e.g., extractive summarization and reading comprehension), models output a variable number of predictions and our interpretation approach will not suffice. Developing interpretation techniques for these types of models is a necessary area for future work.

We apply DKNN to neural models for text classification. This provides a better estimate of model uncertainty—conformity—which we combine with leave-one-out. This overcomes issues stemming from neural network confidence, leading to higher precision interpretations. Most interestingly, our interpretations are supported by the training data, providing insights into the representations learned by a model.

Acknowledgments

Feng was supported under subcontract to Raytheon BBN Technologies by DARPA award HR0011-15-C-0113. JBG is supported by NSF Grant IIS1652666. Any opinions, findings, conclusions, or recommendations expressed here are those of the authors and do not necessarily reflect the view of the sponsor. The authors would like to thank the members of the CLIP lab at the University of Maryland and the anonymous reviewers for their feedback.

References

Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. 2016. Explaining predictions of non-linear classifiers in NLP. In *Workshop on Representation Learning for NLP*.

Jon Louis Bentley. 1975. Multidimensional binary search trees used for associative searching. *Communications of the ACM* .

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

Rich Caruana, Hooshang Kangarloo, John David N. Dionisio, Usha S. Sinha, and David B. Johnson. 1999. Case-based explanation of non-case-based learning methods. *Proceedings of AMIA Symposium* .

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *EMNLP*.

Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv: 1702.08608* .

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2017. HotFlip: White-box adversarial examples for text classification. In *ACL*.

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. In *EMNLP*.

Yarin Gal, Yutian Chen, Roger Frigola, S. Gu, Alex Kendall, Yingzhen Li, Rowan McAllister, Carl Rasmussen, Ilya Sutskever, Gabriel Synnaeve, Nilesh Tripuraneni, Richard Turner, Oriol Vinyals, Adrian Weller, Mark van der Wilk, and Yan Wu. 2016. *Uncertainty in Deep Learning*. Ph.D. thesis, University of Oxford.

Amirata Ghorbani, Abubakar Abid, and James Y. Zou. 2017. Interpretation of neural networks is fragile. *arXiv preprint arXiv: 1710.10547* .

Aristides Gionis, Piotr Indyk, and Rajeev Motwani. 1999. Similarity search in high dimensions via hashing. In *VLDB*.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *ICLR*.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural networks : the official journal of the International Neural Network Society* .

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *ICML*.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *NAACL*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *KDD*.

Heinrich Jiang, Been Kim, and Maya R. Gupta. 2018. To trust or not to trust a classifier. *arXiv preprint arXiv: 1805.11783* .

Been Kim, Cynthia Rudin, and Julie A. Shah. 2014. The bayesian case model: A generative approach for casebased reasoning and prototype classification. In *NIPS*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *EMNLP* .

Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T. Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. 2017. The (un)reliability of saliency methods. *arXiv preprint arXiv: 1711.00867* .

Gary A. Klein. 1989. Do decision biases explain too much. In *Proceedings of the Human Factors and Ergonomics Society*.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *ICML*.

Jiwei Li, Will Monroe, and Daniel Jurafsky. 2016. Understanding neural networks through representation erasure. *arXiv preprint arXiv: 1612.08220* .

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLT*.

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

Anh Mai Nguyen, Jason Yosinski, and Jeff Clune. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL*.

Nicolas Papernot and Patrick D. McDaniel. 2018. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv: 1803.04765* .

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *EMNLP*.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In **SEM@NAACL-HLT*.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should i trust you?": Explaining the predictions of any classifier. In *KDD*.

Glenn Shafer and Vladimir Vovk. 2008. A tutorial on conformal prediction. *JMLR* .

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda B. Viégas, and Martin Wattenberg. 2017. SmoothGrad: removing noise by adding noise. *arXiv preprint arXiv: 1706.03825* .

Richard Socher, A. V. Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *ICML*.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *ICLR*.

Eric Wallace and Jordan Boyd-Graber. 2018. Trick me if you can: Adversarial writing of trivia challenge questions. In *Proceedings of ACL 2018 Student Research Workshop*.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. In *LREC*.

Jake Zhao and Kyunghyun Cho. 2018. Retrieval-augmented convolutional neural networks for improved robustness against adversarial examples. *arXiv preprint arXiv: 1802.09502* .