Improving the Explainability of Neural Sentiment Classifiers via Data Augmentation

Hanjie Chen
Department of Computer Science
University of Virginia
Charlottesville, VA 22904
hc9mx@virginia.edu

Yangfeng Ji
Department of Computer Science
University of Virginia
Charlottesville, VA 22904
yangfeng@virginia.edu

Abstract

Sentiment analysis has been widely used by businesses for social media opinion mining, especially in the financial services industry, where customers' feedbacks are critical for companies. Recent progress of neural network models has achieved remarkable performance on sentiment classification, while the lack of classification interpretation may raise the trustworthy and many other issues in practice. In this work, we study the problem of improving the explainability of *existing* sentiment classifiers. We propose two data augmentation methods that create additional training examples to help improve model explainability: one method with a predefined sentiment word list as external knowledge and the other with adversarial examples. We test the proposed methods on both CNN and RNN classifiers with three benchmark sentiment datasets. The model explainability is assessed by both human evaluators and a simple automatic evaluation measurement. Experiments show the proposed data augmentation methods significantly improve the explainability of both neural classifiers.

1   Introduction

Sentiment analysis is one of the most widely-used applications of natural language processing (NLP) in the financial services industry (Sohangir et al., 2018; Daniel et al., 2017; Li et al., 2018; Chen et al., 2016), where neural sentiment classifiers help enterprises gauge pubic opinion, conduct market research, monitor brand and product reputation, and understand customer experiences (Feldman, 2013; Pang et al., 2002; Liu, 2012). The recent development of neural network modeling has largely boosted the prediction performance (e.g., accuracy) on sentiment classification (Yang et al., 2016; Johnson and Zhang, 2017, 2016; Zhou et al., 2016), while the nonlinearity of neural network models hinders the understanding on predictions. A fair question with no easy answer for neural sentiment classifiers is *why the prediction on this text is positive (or negative)?* Moreover, the lack of explainability on model prediction will raise the issue of trustworthy and fairness of sentiment classifiers in practice (Gilpin et al., 2018; Yu and Kumbier, 2019).

To address the explainability issue of neural classifiers, various approaches have been developed recently to provide model-agnostic explanations on predictions (Ribeiro et al., 2016, 2018; Lundberg and Lee, 2017). Particularly, this work focuses on local explanations, which aims to explain predictions for individual data. The most common way of generating local explanations in sentiment classification is identifying the important part of a text associated with predicted sentiment polarity (Lei et al., 2016; Chen et al., 2018; Nguyen, 2018). For example, a widely-adopted local explanation method LIME (Ribeiro et al., 2016) can identify a set of keywords as an explanation.[1]

---

[1] A brief description of LIME is provided in section 3.

Review	a truly moving experience, and a perfect example of how art when done right can help heal, clarify, and comfort.
Pred. A	Positive
Exp. A	a, moving, can
Pred. B	Positive
Exp. B	perfect, comfort, truly

Table 1: Explanations generated by the local explanation method LIME from two neural sentiment classifiers. The ground-truth sentiment polarity of the text is positive and *both* models give the right prediction.

Table 1 presents an example of movie reviews and the explanations based on two neural sentiment classifiers. Although both classifiers give the right prediction on this example, the explanation A is harder to be *interpreted* than the explanation B, in terms of why the prediction is positive. This difference on the explainability of explanations leads us to trust more on prediction B than A, which will eventually discriminate the practical values of these two sentiment classifiers.

In general, a prediction explanation can be generated by using any local explanation method (Ribeiro et al., 2016; Kindermans et al., 2017; Fong and Vedaldi, 2017; Dabkowski and Gal, 2017). However, the real challenge in practice is that whether an explanation is easy to be interpreted, as demonstrated in Table 1. By noticing the connection and difference between explanations and their explainability, we would like to study the problem on **improving the explainability of neural sentiment classifiers**. We consider this as a learning problem and propose to resolve it with some data augmentation methods. The goal is to increase the explainability of existing neural sentiment classifiers while maintaining similar prediction performance.

In this work, we explore the strategy of using data augmentation to improve the explainability of neural sentiment classifiers. We propose two data augmentation methods: one with a predefined sentiment word list as external knowledge and the other with adversarial examples. Experiments on the two base models and three benchmark datasets show that the proposed methods improve the model explainability with respect to both automatic and human evaluation.

2 Data Augmentation Methods

The basic idea is to teach the models to make predictions based on critical information. In the scenario of sentiment classification, the task is to teach model to make predictions by grasping sentiment words. This section presents two proposed methods for data augmentation and a unified method of using augmented data for training.

2.1 Augmenting via External Knowledge

The first method is called data augmentation with external knowledge (DA-EK). We propose a simple method to create some examples that are similar to training examples with respect to their surface forms, but those examples do not belong to any of the predefined classes $\mathcal{Y}$. To be specific, the augmented examples for sentiment analysis are the examples that are similar to original training examples but have no sentiment polarity, as illustrated in Table 2.

A simple way to create an augmented example $\tilde{x}$ is that, for a given sentence x, removing words $\{x_i\}$ from x if x_i belongs to a predefined sentiment word list. In this work, we use the words listed in the SentiWordNet corpus (Baccianella et al., 2010) and their sentiment polarity scores. For a given sentence x, removing x_i from x if x_i is in the word list will create an augmented example $\tilde{x}$. Table 2 presents two examples of the original text and its augmented counterpart after removing words with clear sentiment polarity. For some simple texts, removing sentiment words will cause their augmented counterparts to be incomplete sentences, which can still be used as augmented data points. For example, if we remove the sentiment word in text `I like this movie`, then the augmented training example is `I this movie`. However, with the training framework proposed in

subsection 2.3, this augmented example will help the model to emphasize the sentiment prediction on the original sentence.

There is a critical distinction between the augmented examples created by DA-EK and the example from the *neutral* class in sentiment classification. In multi-class sentiment classification, there is often a class with an average sentiment score called the neutral class. The major difference is that texts from a neutral class still have sentiment, or at least contains some sentiment words. For example, the movie review `The Cockettes provides a window into a subculture hell-bent on expressing itself in every way imaginable.` is a neutral but not augmented example. With words like `hell-bent` and `imaginable`, it shows sentiment inclination of this movie review even though it is not strong. To construct an augmented example from this text, the proposed method still needs to remove the sentiment words. Empirical results show that adding neutral examples can only lead to a minor improvement on explainability.

2.2 Augmenting with Adversarial Examples

This method is called data augmentation with adversarial examples or DA-ADV. We adopt the method proposed by Alzantot et al. (2018) to generate adversarial examples, which may have similar surface forms and semantic meanings to training examples. To be specific, this method aims to minimize the number of modified words between the original and adversarial examples, and maintain semantic and syntactic similarity by substituting only a few synonyms. A well-known challenge on generating adversarial examples in text data is that texts are discrete, which causes the difficulty in generating adversarial examples by using the popular gradient-based methods (Goodfellow et al., 2014; Kurakin et al., 2016; Madry et al., 2017). Alzantot et al. (2018) developed an attack algorithm via genetic algorithms. In each generation, a group of candidate examples are generated by substituting synonyms, and those most fit within the context surrounding are selected by the Google 1-billion words language model (Chelba et al., 2013). The candidates that can successfully attack the model to flip prediction polarity are adversarial examples. Like many other adversarial attack methods, there is a budget about how many words can be replaced. Beyond that budget limit will cause a fail attack. In our case, it means not every text can get an adversarial example.

As adversarial examples can flip model predictions, the replaced words from original texts must be critical to sentiment prediction. Similar to the previous data augmentation method, we can construct augmented examples by taking the replaced words as the sentiment words in DA-EK.

Comparison. Table 2 presents some examples of two data augmentation methods. With DA-EK, we have a high-precision method for data augmentation. If any word in a text matches one entry in the SentiWordNet, then it is very likely to be a sentiment word. However, the word list in the SentiWordNet is predefined and definitely not comprehensive. The missing sentiment words imply DA-EK could be a data augmentation method with low recall. With DA-ADV, we have a low-precision method for data augmentation. Words identified by adversarial attacks can be sentiment words or simply can be non-sentiment words that are sensitive to neural sentiment classifiers. Besides, finding adversarial examples is very time consuming, as further explained in section 4. But DA-ADV has the potential to extend this method to other text classification tasks, where we do not have a pre-defined word list.

2.3 Learning with Augmented Examples

We extend the training set $\mathcal{D} = \{(\boldsymbol{x}^{(k)}, y^{(k)})\}$ as by adding the augmented examples $\{(\widetilde{\boldsymbol{x}}^{(k')}, \text{AUG})\}$ generated by either DA-EK or DA-ADV and extend it as $\widetilde{\mathcal{D}} = \mathcal{D} \cup \{(\widetilde{\boldsymbol{x}}^{(k')}, \text{AUG})\}$. Similarly, the label set $\mathcal{Y}$ is also extended as $\widetilde{\mathcal{Y}} = \mathcal{Y} \cup \{\text{AUG}\}$. Note that, the proposed methods only create augmented examples for the training set and development set. No modification is on the test set.

Once we have the extended training and development set, learning a neural sentiment classifier with data augmentation is straightforward. Specifically, we optimize the following loss function

$$\underset{\boldsymbol{\theta}}{\text{argmax}} \sum_{\widetilde{\mathcal{D}}} \mathcal{L}(\hat{y}^{(k)}, y^{(k)}), \tag{1}$$

to achieve the best prediction accuracy on the *augmented* development set, where $\hat{y}^{(k)}$ is decoded from the decision function defined in Equation 5 with extended label set $\widetilde{\mathcal{Y}}$, $\mathcal{L}(\cdot, \cdot)$ is the cross-

Original text	the only problem is that, by the end, no one in the audience or the film seems to really care
DA-EK	the only that , by the end, one in the audience or the film seems to care
Adversarial example	the only difficulty is that, by the end, no one in the audience or the movie seems to really caring
DA-ADV	the only is that, by the end, no one in the audience or the seems to really
Original text	michel piccoli's moving performance is this films reason for being
DA-EK	michel piccoli's this films reason for being
Adversarial example	michel piccoli's moving play is this movie reason for being
DA-ADV	michel piccoli's moving is this reason for being

Table 2: Some examples of the augmented data created by DA-EK and DA-ADV.

entropy loss, and $\boldsymbol{\theta}$ is the collection of parameters, which is the same as the base model. During test, with no augmented example, the trained neural classifier simply ignores any prediction on the label AUG and picks the label from $\mathcal{Y}$ that maximizes the decision value in Equation 5. More information of the neural classifiers used in our experiments is provided in **??**.

3   Experimental Setup

This section describes the experimental setup used in this work. We test the proposed data augmentation methods with two neural sentiment classifiers, a convolutional neural network in (Kim, 2014, CNN) and a recurrent neural network with LSTM (Hochreiter and Schmidhuber, 1997, RNN), on three benchmark datasets, SST (Socher et al., 2013), MR (Pang and Lee, 2005), and IMDB (Maas et al., 2011). Local explanations were generated from LIME (Ribeiro et al., 2016) with model predictions and the cosine similarity method based on text representations. More implementation details are in Appendix A.

3.1   Local Explanation Generation

To generate local explanations, we adopt the LIME proposed by Ribeiro et al. (2016) to generate explanations on model predictions. Besides, we also suggest another way of generating local explanations based on the cosine similarity between word representations and text representations.

3.1.1   LIME with model predictions

The basic idea of the LIME is that, for a given example $\boldsymbol{x}$, it finds an explanation based on a locally linear approximation $\boldsymbol{g}(\boldsymbol{z}^{(l)}, y)$ of the decision function $\boldsymbol{h}(\boldsymbol{z}, y)$, in which $\boldsymbol{z}$ is a perturbation of $\boldsymbol{x}$ obtained by subsampling the words from $\boldsymbol{x}$. Given a set of subsamples $\{\boldsymbol{z}^{(l)}\}$ from $\boldsymbol{x}$, the loss function of the LIME is defined as

$$L(\boldsymbol{h}, \boldsymbol{g}) = \sum_{i=1}^{q} D_{\boldsymbol{x}, \boldsymbol{z}^{(l)}} (\boldsymbol{h}(\boldsymbol{z}^{(l)}, y) - \boldsymbol{g}(\boldsymbol{z}^{(l)}, y))^2, \tag{2}$$

where the linear approximation function $\boldsymbol{g}$ is usually defined as $\boldsymbol{g}(\boldsymbol{z}, y) = \boldsymbol{w}_y^\top \boldsymbol{z}$. $D_{\boldsymbol{x}, \boldsymbol{z}^{(l)}}$ measures the similarity between $\boldsymbol{x}$ and $\boldsymbol{z}^{(l)}$,

$$D_{\boldsymbol{x}, \boldsymbol{z}^{(l)}} = \exp\left(\frac{-d(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{z}^{(l)}))^2}{\sigma^2}\right), \tag{3}$$

with $d(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{z}^{(l)})$ as the cosine distance between the latent representations of $\boldsymbol{x}$ and $\boldsymbol{z}^{(l)}$, as suggested in (Ribeiro et al., 2016).

Optimizing Equation 2 will try to match the decision values from linear approximation $\boldsymbol{g}(\boldsymbol{z}^{(l)}, y)$ with $\boldsymbol{h}(\boldsymbol{z}^{(l)}, y)$ and also produce a set of linear weights $\{\boldsymbol{w}_y\}_{y \in \mathcal{Y}}$ associated with $\mathcal{Y}$. The values of

4

$\{\boldsymbol{w}_{y,i}\}$ indicate the importance of $\{\boldsymbol{x}_i\}$. If the predicted label is $\hat{y}$, then top t words according to $\{\boldsymbol{w}_{\hat{y},i}\}$ will be selected as an explanation of $\boldsymbol{x}$ on the corresponding prediction.

3.1.2 Cosine similarity on text representations

For a given text $\boldsymbol{x}$, the basic idea of using cosine similarity generating explanations is to measure the similarity between its text representation $\boldsymbol{f}(\boldsymbol{x})$ and word representations $\boldsymbol{f}(\boldsymbol{x}_i)$, where $\boldsymbol{x}_i$ is the embedding of the i-th word in text $\boldsymbol{x}$. In this way, we can find the most similar words with respect to the text representation $\boldsymbol{f}(\boldsymbol{x})$, and choose the top t words as an explanation. The underlying assumption of this idea is that, if a text representation $\boldsymbol{f}(\boldsymbol{x})$ could facilitate sentiment prediction, the sentiment polarity indicated by the top t similar words should be consistent with its overall sentiment polarity.

To compute cosine similarity, we first need to map all the word embeddings $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ into the text representation space using $\boldsymbol{f}(\cdot)$. Then, the similarity between a text and the i-th word within this text is measured by the cosine value of these two vectors,

$$\text{cos-sim}(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{x}_i)) = \frac{\langle \boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{x}_i) \rangle}{\|\boldsymbol{f}(\boldsymbol{x})\|_2 \cdot \|\boldsymbol{f}(\boldsymbol{x}_i)\|_2}. \tag{4}$$

After applying Equation 4 to every word in text $\boldsymbol{x}$, then we pick the top t words with respect to their cosine similarities as an explanation of $\boldsymbol{f}(\boldsymbol{x})$.

4 Experiments

Although there are different ways to evaluate prediction explanations as suggested in prior work (Ribeiro et al., 2016; Gilpin et al., 2018), the explainability of explanations should be the most of important criterion. As argued by Gilpin et al. (2018), a good explanation should be easily interpretable and "*simple enough for a person to understand using a vocabulary that is meaningful to the user*". For sentiment classification, as demonstrated in the running example (Table 1), a good explanation on sentiment prediction should be easy enough for a human user to understand together with the prediction. Following this intuition, we define the explainability measurement for both automatic evaluation and human evaluation.

4.1 Automatic Evaluation

Our automatic evaluation method measures the explainability of an local explanation (consisting of a set of keywords) by predicting its sentiment polarity and comparing with the model prediction. Specifically, for each keyword in an explanation, we retrieve its sentiment scores from the Senti-WordNet. SentiWordNet offers three scores for a sentiment word: a positive score, a negative score, and a neutral score. For the word `truly`, its positive score is 0.625, negative score is 0 and neutral score is 0.375, which indicates that it is a word with positive polarity. On the other hand, the positive score of word `a` is 0 and its neutral score is 1. With the sentiment scores of these words, the overall scores of an explanation is the accumulation of the sentiment scores of its keywords.

Consider the local explanations in Table 1, the sentiment scores of explanation A with respect to the positive sentiment polarity is 0 and the positive socre of explanation B is 0.625. Under this simple automatic evaluation measurement, the explanation A is easier to be interpreted than explanation B, which is consistent with our expectation.

To quantificationally evaluate prediction explanations, we propose the *coherence score* defined as follow: for a given test example, depending on the sentiment polarities predicted by the model, indicated by the explanation and the ground truth, it will be counted as a coherent case, if it satisfies one of the two conditions:

- Condition 1: if the sentiment polarity indicated by the explanation is not NONE and is consistent with the model prediction; or
- Condition 2: if the sentiment polarity indicated by the explanation is NONE and the model prediction is not the same as the ground truth.

The coherence between the model prediction and its prediction in condition 1 is obvious. About condition 2, we consider that an explanation with no sentiment polarity is also coherent with a

Dataset	Model	Accuracy	Coherence	
			LIME	Cos-Sim
SST	CNN	0.85	0.65	0.58
	CNN-EK	0.85	0.70	0.60
	CNN-ADV	0.85	0.68	0.59
	RNN	0.84	0.64	0.61
	RNN-EK	0.84	0.66	0.62
MR	CNN	0.81	0.63	0.50
	CNN-EK	0.80	0.66	0.55
	CNN-ADV	0.80	0.65	0.55
	RNN	0.80	0.64	0.59
	RNN-EK	0.80	0.65	0.60
IMDB	CNN	0.90	0.76	0.23
	CNN-EK	0.90	0.80	0.53
	CNN-ADV	0.90	0.78	0.47
	RNN	0.87	0.78	0.74
	RNN-EK	0.87	0.81	0.78

Table 3: The classification and explainability evaluation results of different models on SST, MR and IMDB. The models trained with augmented data from DA-EK are named with -EK. The model trained with the augmented data from DA-ADV is named with -ADV.

wrong prediction. Intuitively, an explanation with no sentiment polarity explains why the prediction is wrong. For a collection of explanations, the coherence score is the ratio of the number of coherent cases to the total number of instances.

4.1.1 Results

Table 3 shows the prediction accuracies and coherence scores of different models on the all three datasets. As indicated in the third column, the models trained with augmented data, including both DA-EK and DA-ADV, maintain the prediction accuracies comparing to their counterparts. This observation matches our expectation that data augmentation for improving explainability should not hurt prediction accuracy.

More important, all models trained with augmented data outperform the base models with respect to the coherence score (column 4 and 5). Comparing the coherence scores with the same base model and the same dataset, we found that, in most of the cases, both data augmentation methods help improve the coherence score, regardless which explanation generation we use. Comparing the scores across multiple datasets and models, we also notice that the improvement on LIME-based explanations has a smaller variance than the explanations generated by the cosine similarity method. We suspect that this is because local explanations are always tied with model predictions, while the cosine similarity method only use text representations to generate explanations.

As shown in the experiments with the CNN-ADV, data augmentation with adversarial examples does provide some benefit to improve the coherence scores of the CNN model on all of the three datasets. The state-of-the-art method (Alzantot et al., 2018) generating adversarial examples can be extended to other neural calssifiers (e.g. RNN) and text classification tasks in the future work.

4.2 Human Evaluation

In subsection 4.1, we propose the coherence score and use it to automatically evaluate the local explanations. Even though it is easy to compute, the major limitation is from the pre-defined list of sentiment words. For a specific test example, this evaluation method will fail if the sentiment words in the generated explanation are not the SentiWordNet word list. Furthermore, as discussed in the beginning of this section, explainability is about whether an explanation is understandable to human users. Human evaluation is necessary if we would like to measure the explainability improvement.

Dataset	Model	Human Evaluation	Automatic Evaluation
SST	CNN	0.85	0.56
	CNN-EK	0.92	0.63
MR	CNN	0.84	0.55
	CNN-EK	0.90	0.60

Table 4: Human and automatic evaluation results on the sets of the SST and MR datasets. The augmented examples are from DA-EK and the explanations are generated by LIME.

Besides the human evaluation results can provide further justification of the coherence scores from automatic evaluation.

4.2.1 Evaluation setup

To conduct a human evaluation task, we random pick 100 test examples from the SST and MR datasets. Explanations of these examples are generated by LIME based on the CNN and CNN-EK models. We have 7 graduate students with proficient English skills as volunteers to evaluate the quality of these explanations.

With a given test example with an explanation pair generated from the CNN and CNN-EK models respectively, a human evaluator needs to a two-step evaluation. First, for each explanation, the human evaluator needs to analyze whether it can interpret the corresponding prediction, and mark with a score ("1" for coherent, "0" for incoherent) according to the two conditions in subsection 4.1, only with the evaluator himself to give the sentiment polarity of the explanation. Then, for the explanation pair, the evaluator will be asked to pick which one better explains the corresponding model prediction. Note that, two explanations within each pair are presented to our human evaluators randomly to eliminate any possible bias.

Finally, the human evaluation score is calculated as the ratio of the sum of the scores to the number of examples. We also calculate the coherence scores on the 100 test examples and compare them with the human evaluation scores.

4.2.2 Results

Table 4 presents both the human evaluation scores and also the coherence scores. On both datasets, human evaluation scores indicate that data augmentation with additional examples improves the explainability of CNN. As shown in Table 5, the explanations from CNN-EK are more interpretable and the sentiment polarity indicated by these two explanations are clear. In addition, the comparison between the human evaluation and the automatic evaluation also shows the coherence scores are positively correlated with the human evaluation scores, which provides a justification for our automatic evaluation measurement.

We also notice that the coherence scores are constantly lower than the human evaluation scores, even though the computations of these two scores are similar. One possible reason is that the pre-defined sentiment word list from the SentiWordNet is not comprehensive enough, while human evaluators can always tell which explanation is better than the other.

5 Related Work

Deep learning has demonstrated success on text classification, including sentiment analysis. Along with the increase of neural network models, there is a growing demanding on building explainable models (Gilpin et al., 2018; Guidotti et al., 2018; Murdoch et al., 2019). Particularly in sentiment classification, prior work on recommendation systems with sentiment analysis shows that recommendations with explanations have more influential impact on users behavior (Zhang et al., 2014; Zhang, 2015). Similar work has also been shown in other application domain of sentiment analysis. For example, Luo et al. (2018) demonstrate an explainable neural network model can uncover informative clues related users preference in financial sentiment analysis.

Input text	Models	Prediction	Keywords	Indication
[Pos] at about 95 minutes, treasure planet maintains a brisk pace as it races through the familiar story	Cnn	Positive	treasure, pace, a	None
	Cnn-Ek	Positive	brisk, treasure, familiar	Positive
[Neg] unfortunately, they're sandwiched in between the most impossibly dry account of kahlo's life imaginable	Cnn	Negative	dry, 're, sandwiched	None
	Cnn-Ek	Negative	unfortunately, dry, account	Negative

Table 5: Examples of the explanations generated by LIME from the Cnn and Cnn-Ek models, where the ground truth of each input text is marked in front of it as "Pos" or "Neg".

This work focus on local explanations, which are generated based on individual model predictions. Among the local explanation methods, LIME (Ribeiro et al., 2016) is probably the most popularly used method, due to its model-agnostic feature. Some other model agnostic explanation methods include SHAP (Lundberg and Lee, 2017) and MAPLE (Lundberg and Lee, 2017), which we will leave for future work. Besides, there are model-based local explanation methods, for example, using saliency maps from gradient information (Li et al., 2016) or attention weights in some neural attention models (Bahdanau et al., 2015). However, both of them are criticized about their ability of generating explanations (Adebayo et al., 2018; Jain and Wallace, 2019).

As mentioned in section 1, we differentiate the explainability of the local explanations from the local explanations themselves. As discussed in (Gilpin et al., 2018), the explainability of an explanation is about whether it is easy for humans to interpret or understand. This distinction emphasizes the importance of human evaluation and helps us design the automatic evaluation method in this work. For general discussion on explainability, we refer the readers to (Guidotti et al., 2018; Gilpin et al., 2018; Murdoch et al., 2019).

To improve the explainability of neural sentiment classifiers, we propose two data augmentation methods to add more training examples. Prior work on data augmentation for text classification (Kobayashi, 2018; Wei and Zou, 2019) mainly focuses more on improving prediction performance. Adversarial examples, as a particular category of augmented data, are mainly used to enhance model robustness (Li et al., 2017; Sun et al., 2018). With the two proposed methods, this work demonstrates a novel way of constructing and using augmented examples.

6 Conclusion

In this paper, we showed that the explainability of neural sentiment classifiers can be improved by training with augmented data. We proposed two data augmentation methods by employing a predefined word list and adversarial examples respectively. In this work, we focused on the explainability of local explanations, which were generated by LIME and the cosine similarity method. Then, the improvement of model explainability was assessed with both automatic evaluation and human evaluation. Experiments showed that the proposed data augmentation methods could successfully improve the model explainability.

References

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., and Kim, B. (2018). Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, pages 9525–9536.

Alzantot, M., Sharma, Y., Elgohary, A., Ho, B.-J., Srivastava, M., and Chang, K.-W. (2018). Generating natural language adversarial examples. *arXiv preprint arXiv:1804.07998*.

Baccianella, S., Esuli, A., and Sebastiani, F. (2010). Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *ICLR*.

Chelba, C., Mikolov, T., Schuster, M., Ge, Q., Brants, T., Koehn, P., and Robinson, T. (2013). One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*.

Chen, H., Sun, M., Tu, C., Lin, Y., and Liu, Z. (2016). Neural sentiment classification with user and product attention. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 1650–1659.

Chen, J., Song, L., Wainwright, M. J., and Jordan, M. I. (2018). Learning to explain: An information-theoretic perspective on model interpretation. *arXiv preprint arXiv:1802.07814*.

Dabkowski, P. and Gal, Y. (2017). Real time image saliency for black box classifiers. In *Advances in Neural Information Processing Systems*, pages 6967–6976.

Daniel, M., Neves, R. F., and Horta, N. (2017). Company event popularity for financial markets using twitter and sentiment analysis. *Expert Systems with Applications*, 71:111–124.

Feldman, R. (2013). Techniques and applications for sentiment analysis. *Communications of the ACM*, 56(4):82–89.

Fong, R. C. and Vedaldi, A. (2017). Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3429–3437.

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., and Kagal, L. (2018). Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 80–89. IEEE.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., and Pedreschi, D. (2018). A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):93.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Jain, S. and Wallace, B. C. (2019). Attention is not explanation. *CoRR*, abs/1902.10186.

Johnson, R. and Zhang, T. (2016). Supervised and semi-supervised text categorization using lstm for region embeddings. *arXiv preprint arXiv:1602.02373*.

Johnson, R. and Zhang, T. (2017). Deep pyramid convolutional neural networks for text categorization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 562–570.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Kindermans, P.-J., Schütt, K. T., Alber, M., Müller, K.-R., Erhan, D., Kim, B., and Dähne, S. (2017). Learning how to explain neural networks: Patternnet and patternattribution. *arXiv preprint arXiv:1705.05598*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kobayashi, S. (2018). Contextual augmentation: Data augmentation by words with paradigmatic relations. *arXiv preprint arXiv:1805.06201*.

Kurakin, A., Goodfellow, I., and Bengio, S. (2016). Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*.

Lei, T., Barzilay, R., and Jaakkola, T. (2016). Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*.

Li, J., Chen, X., Hovy, E., and Jurafsky, D. (2016). Visualizing and understanding neural models in nlp. In *Proceedings of NAACL-HLT*, pages 681–691.

Li, X., Bing, L., Lam, W., and Shi, B. (2018). Transformation networks for target-oriented sentiment classification. *arXiv preprint arXiv:1805.01086*.

Li, Y., Cohn, T., and Baldwin, T. (2017). Robust training under linguistic adversity. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 21–27.

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774.

Luo, L., Ao, X., Pan, F., Wang, J., Zhao, T., Yu, N., and He, Q. (2018). Beyond polarity: Interpretable financial sentiment analysis with hierarchical query-driven attention. In *IJCAI*.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. (2011). Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pages 142–150. Association for Computational Linguistics.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., and Yu, B. (2019). Interpretable machine learning: definitions, methods, and applications. *arXiv preprint arXiv:1901.04592*.

Nguyen, D. (2018). 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)*, pages 1069–1078.

Pang, B. and Lee, L. (2005). Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 115–124. Association for Computational Linguistics.

Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144. ACM.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2018). Anchors: High-precision model-agnostic explanations. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Sohangir, S., Wang, D., Pomeranets, A., and Khoshgoftaar, T. M. (2018). Big data: Deep learning for financial sentiment analysis. *Journal of Big Data*, 5(1):3.

Sun, S., Yeh, C.-F., Ostendorf, M., Hwang, M.-Y., and Xie, L. (2018). Training augmentation with adversarial examples for robust speech recognition. *arXiv preprint arXiv:1806.02782*.

Wei, J. W. and Zou, K. (2019). Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., and Hovy, E. (2016). Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

Yu, B. and Kumbier, K. (2019). Three principles of data science: predictability, computability, and stability (pcs). *arXiv preprint arXiv:1901.08152*.

Zhang, Y. (2015). Incorporating phrase-level sentiment analysis on textual reviews for personalized recommendation. In *Proceedings of the eighth ACM international conference on web search and data mining*, pages 435–440. ACM.

Zhang, Y., Lai, G., Zhang, M., Zhang, Y., Liu, Y., and Ma, S. (2014). Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 83–92. ACM.

Zhou, P., Qi, Z., Zheng, S., Xu, J., Bao, H., and Xu, B. (2016). Text classification improved by integrating bidirectional lstm with two-dimensional max pooling. *arXiv preprint arXiv:1611.06639*.

Dataset	C	L	N
SST	2	19	9613
MR	2	20	10662
IMDB	2	268	50000

Table 6: Summary statistics for the datasets, where C is the number of classes, L is average sentence length, and N is dataset size.

A Implementation Details

Datasets We use three sentiment benchmark datasets for evaluation. Summary statistics of the datasets are in Table 6 .

- **SST.** This dataset was proposed in (Socher et al., 2013) for sentence-level sentiment classification. We used the SST-2, which is the 2-class version of this dataset. There are 6,920 examples in the training set, 872 examples in the development set, and 1,821 examples in the test set. For data augmentation with DA-EK, additional 1,624 and 229 augmented examples were added to the training and development sets respectively. With DA-ADV, 4,885 and 539 augmented examples were added to the training and development sets respectively.

- **MR.** This dataset was proposed by Pang and Lee (2005). These reviews in this dataset were divided into 9,596 training and 1066 test examples. In our experiments, 90% of the training examples are used for training, and the rest is used as development set. With DA-EK, we added additional 4,318 and 480 augmented examples to the training and development sets respectively.

- **IMDB.** This dataset was proposed by Maas et al. (2011). These reviews in this dataset were divided into 25,000 training and 25,000 test examples. We split 90% of the training examples for training, and the rest as the development set. With DA-EK, additional 11,250 and 1,250 augmented examples were added to the training and development sets respectively.

Neural Sentiment Classifiers In this work, we use a convolutional neural network in (Kim, 2014, CNN) and a recurrent neural network with LSTM (Hochreiter and Schmidhuber, 1997, RNN) as our baseline models.

The CNN consists of an input layer that takes word embeddings as inputs, a convolutional layer followed by a max-pooling layer for composing word embeddings into text representations, and a softmax layer for classification. For a given text $\boldsymbol{x}$, $\boldsymbol{f}(\cdot)$ denotes the representation function in CNN, which maps $\boldsymbol{x}$ into a d-dimensional numeric vector $\boldsymbol{f}(\boldsymbol{x})$ as text representation. The decision function is defined as

$$\boldsymbol{h}(\boldsymbol{x}, y) = \boldsymbol{u}_y^\top \boldsymbol{f}(\boldsymbol{x}), \tag{5}$$

where $y \in \mathcal{Y}$ is the class label, and $\mathbf{u}_y \in \mathbb{R}^d$ is the corresponding classification weight vector. For prediction, we use $\hat{y} = \text{argmax}_y \boldsymbol{h}(\boldsymbol{x}, y)$.

The RNN consists of uni-directional one-layer LSTM. For a given text, the last hidden state of this RNN is used as the text representation $\boldsymbol{f}(\boldsymbol{x})$. The same decision function defined in Equation 5 is employed for sentiment prediction. The more specific configuration and some implementation details of these two models are provided in Appendix A.

Even though the main focus of this work is on model explainability, the prerequisite is to match the classification performance in prior work with the similar model architectures. Here are some implementation details that we adopted from prior work (Kim, 2014): For both the CNN, CNN-EK and CNN-ADV, we used a single convolutional layer with filters of the window sizes ranging from 3 to 5. For both RNN and RNN-EK, we used a single layer LSTM. For all of the models, the input parameters were initialized with the 300-dimensional pretrained word embeddings (Mikolov et al., 2013, word2vec) and all other parameters were randomly initialized with the default method in PyTorch. Hyperparameters, including kernel size (for CNN only), hidden size (for RNN only), learning rate, minibatch size, etc., were tuned separately on the development set for different datasets. We used Adam (Kingma and Ba, 2014) to update the parameters.

LIME Setup For LIME, the number of subsamples was set to 600, σ^2 was 10, and each local linear model was trained over 50 epochs. For adversarial training, the attacker is limited to 20 iterations, and some other hyper-parameters are fixed: nearest neighbors was 8, top kept words was 4, and the maximum percentage of allowed change to the text was 20%.