EVALUATION OF LSTM EXPLANATIONS IN SENTIMENT CLASSIFICATION TASK

A Thesis

by

JICHENG LU

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Anxiao Jiang
Committee Members,	Ruihong Huang
	Xiaoning Qian
Head of Department,	Scott Schaefer

December 2020

Major Subject: Computer Science

ABSTRACT

Deep learning techniques produce impressive performance in many natural language processing tasks. However, it is still difficult to understand what the neural network learned during training and prediction. Recently, Explainable Artificial Intelligence (XAI) is becoming a popular technique to interpret deep neural networks. In this work, we extend the existing Layer-wise Relevance Propagation (LRP) framework and propose novel strategies on passing relevance through weighted linear and multiplicative connections in LSTM. Then we evaluate these explainable methods on a bidirectional LSTM classifier by performing four word-level experiments: sentiment decomposition, top representative words collection, word perturbation and case study. The results indicate that the ϵ-LRP-all method outperforms other methods in this task, due to its ability to generate reasonable word-level relevance, collect reliable sentiment words and detect negation patterns in text data. Our work provides an insight on explaining recurrent neural networks and adapting explainable methods to various applications.

DEDICATION

To my mother Mrs. Yueping Peng, my father Mr. Yayong Lu, my family friend Mrs. Eileen Bei and all my family and friends who have supported me during my study at Texas A&M University.

ACKNOWLEDGMENTS

I would like to thank my parents and my family friend Mrs. Eileen Bei who gave me strong physical and mental support during my pursuit of this Master degree. Thanks for them to share valuable life experience with me.

Thanks for my advisor Dr. Anxiao Jiang. He always gives me professional academic advice and unique insights to my research. I would also like to thank my committee members, Dr. Ruihong Huang and Dr. Xiaoning Qian, who have offered me guidance and support both in their courses and in my research.

Moreover, I would like to express my gratitude to Dr. Leila Arras and Dr. Alexander Binder, who have gave me valuable advice when I met difficulties in my research.

Thanks to my colleagues, the faculties and the staffs in the Computer Science and Engineering Department for offering me favoring experience during my study at Texas A&M University.

Thanks to my two roommates, Mr. Hanbin Hu and Mr. Tingjui Chang, for sharing their experience and life support with me.

Thanks to my Friend Mr. Ryan Gearner, Mr. Ben Burleson and the Lamkins family, who have been supporting me since I was in College Station.

Thanks to everyone in the college who gave me support in any aspect in my life.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This research was supported by a thesis committee consisting of Dr. Anxiao Jiang and Dr. Ruihong Huang of the Department of Computer Science and Engineer and Dr. Xiaoning Qian of the Department of Electrical and Computer Engineering.

Funding Sources

No outside funding was received for the research and writing of this document.

NOMENCLATURE

AI	Artificial Intelligence
XAI	Explainable Artificial Intelligence
NLP	Natural Language Processing
MLP	Multilayer Perceptron
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
LSTM	Long Short-Term Memory
Bi-LSTM	Bi-directional Long Short-Term Memory
LRP	Layer-wise Relevance Propagation
ReLU	Rectified Linear Units
MSE	Mean Square Error
SGD	Stochastic Gradient Descent
SVM	Support-Vector Machine
SA	Sensitivity Analysis
ϵ-LRP-all	LRP with ϵ-rule and $zero - one$ distribution
ϵ-LRP-$relative$	LRP with ϵ-rule and $relative$ distribution
ϵ-LRP-abs	LRP with ϵ-rule and $absolute$ distribution
β-LRP-all	LRP with β-rule and $zero - one$ distribution
abs-LRP-all	LRP with abs-rule and $zero - one$ distribution
R_i	Relevance of neuron i
$R_{i \leftarrow j}$	Relevance message from neuron j to neuron i
z_i	Value of neuron i

ϵ	Stabilizer
W_{ij}	Weight value that connects neuron i and neuron j
b_i	Bias of neuron i
δ	Bias factor
N	Number of neurons at lower layer that connect to a neuron at upper layer
z_{ij}	Weighted connection between neuron i and neuron j
z_{ij}^{+}	Positive connection between neuron i and neuron j
z_{ij}^{-}	Negative connection between neuron i and neuron j
$count_{pos}$	Number of positive connections
$count_{neg}$	Number of negative connections

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

In this chapter, we introduce the background of Explainable Artificial Intelligence (XAI), and point out the motivation and current challenges in this area. Next, we state our research problem and the objectives we plan to achieve. We list the thesis outline at the end of this chapter.

1.1 Motivation

Artificial Intelligence (AI) is becoming the center of many activities in this new information era. Among the AI technologies, machine learning techniques have achieved impressive performance when solving complex tasks in many fields, such as medicine [6], finance [7] and management [8]. Because of their strong learning capability, we can build create the models directly from the data by a machine learning algorithm. In other words, machine learning models have been considered as black boxes. That is, it is difficult for people to understand how these models make their decisions or how the outcomes are related to the input elements through huge number of variables. Therefore, there is a limitation in the effectiveness and validity of machine learning.

In the recent decade, deep learning, as a subset of machine learning in artificial intelligence, has achieved the state-of-the-art performance in many areas, e.g., self-driving vehicles [9], robotics [10], speech recognition [11], etc. It imitates the human behaviors to process input data and fulfill different tasks, such as recognizing visual objects and voice, making classifications and translating languages. One of the advantages of deep learning over traditional machine learning is that deep learning is able to automatically learn features from lower level to higher level through layers of neural networks while traditional machine learning requires feature extraction and feature engineering by experts to reduce the data complexity [12]. Moreover, deep learning is capable of handling larger amount of data with even higher accuracy, which fulfills the human needs in the Big Data era (Figure 1.1).

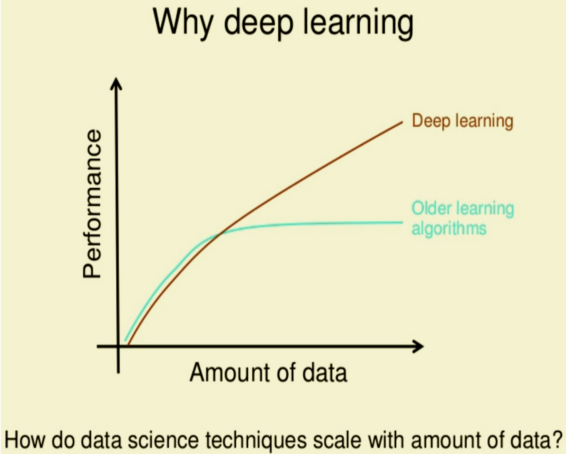

Figure 1.1: Comparison between deep learning and machine learning, reprinted from [1].

However, given all the advantages of deep learning, there is still concern to understand how deep learning models can make correct decisions while the huge number of model parameters are interacting with each other under iterations. For example, given a news article and a deep learning model classifies it as sports topic, we want to know the words in the original text that directly let the model make this decision. Therefore, there is an emerging need to understand how solutions or decisions are achieved by deep learning models.

Recently, Explainable Artificial Intelligence, also called XAI, becomes a new technology in explaining deep learning systems. It is based on the following needs:

- First, we need to find the causality between the input and output. For example, when performing a sentiment analysis task, we need to justify why a deep learning model predicts a movie review as positive or negative. To be more specific, we need to use the model to point out the words that have positive contributions to the final prediction. In this case, we are able to explicit the complex data patterns that are not easily detected by human.

- Second, explainable model can increase the mutual trust between users and deep learning systems. For example, many hospitals or medicine research centers are using deep learning models to help diagnose diseases, such as Alzheimer's disease [13], but so far, human doctors still need to get involved and check the results before selecting treatment methods. If the

model can explain itself, both doctors and patients can understand the rationale behind its conclusions.

- Lastly, XAI gives us a better understanding of the model, which enables us to detect model weakness, improve model structures and even learn new insights.

In reality, XAI has already attracted attention from both industry and academia. Organizations, such as Defense Advanced Research Projects Agency (DARPA), have launched a project of XAI [14]. Some high-tech companies, such as Google, started People +AI Research (PAIR) project [1]. The project aims to interpret AI recommendations and predictions through interactions, examples and displaying model confidence. Facebook has introduced a library called Captum to explain the decisions made by deep neural networks [2]. It allows researchers to understand AI decisions in multimodal environment which combines text, audio, video and images. People can also use Captum to understand the contributions of each neuron or layer to the final output. Amazon SageMaker [3], the platform of building, training machine learning models and applying AI services, has made solid improvement in machine learning explainability based on SHAP and there are more plans in this crucial area.

In academia, researchers develop lots of strategies and methods to explain the model behavior, reveal the inner structure and create visual interfaces that are understandable by users [15, 16, 17]. In deep learning area, XAI has been adapted for explaining Deep Neural Networks (DNN), including multi-layer neural networks [18, 19, 20], Convolutional Neural Networks (CNN) [21, 22, 23] and Recurrent Neural Networks (RNN) [24, 25, 26]. In this work, we focus on explaining RNN, especially the Long Short-Term Memory (LSTM). RNN has been extensively used for processing the sequential data (such as text or time-series signals), with a notable implementation in Natural Language Processing (NLP). Although great achievements have been made in CNN and computer

[1] https://pair.withgoogle.com/guidebook/

[2] https://venturebeat.com/2019/10/10/facebooks-captum-brings-explainability-to-machine-learning/

[3] https://www.cmswire.com/information-management/amazons-ai-leadership-advances-at-reinvent-2019/

vision [27, 28, 29], there are few contributions made for interpreting RNN. In general, these works can be divided into two categories:

- Explain and understand what an RNN has learned.
- Explain and modify RNN architectures to provide new learning framework.

Among the first category, Ding et al. [30] use layer-wise relevance propagation method to study the contribution of contextual words to any intermediate hidden state. Karpathy et al. [25] propose a qualitative visualization method to interpret that LSTM models can learn long-range interactions on word sequences. Besides the work explaining RNN, Arras et al. [31] explore the inner structure of LSTM and propose a simplified LSTM that can solve the same task with less memory cells.

Given the fact that deep learning will keep developing in the next a few decades, we believe that explaining deep learning model will become an important aspect in both industry and academia. Moreover, there are fewer contributions in explaining RNN than CNN. It is worth studying the interpretability of RNN and LSTM.

1.2 Challenge

In early NLP works, researchers used to build their systems based on language features or rules, such as semantic relation, word frequency, document frequency, syntactic types, lexical classes. These features can be easily understood by human, and we can observe the importance of these features to explain what the NLP models can learn. However, it is rather difficult to understand what is happening inside an end-to-end deep neural network. For example, when we conduct sentiment analysis of movie reviews, we usually input the text words and go through the word embedding layer and the recurrent neural network until we obtain the final output, e.g., positive or negative. However, we cannot understand what happens inside the model and the rationale that the model's decision is based on. This requires us to open the black box, check the hidden nodes, and propagate from output label back to the input words. Although projecting word embedding to

lower-dimensional space [32] and the attention mechanism [33] can explain LSTM models locally, interpreting the entire neural network from output to input still remains a challenge.

1.3 Problem Statement

In the Natural Language Processing area, there are many well-known applications, such as name entity recognition, speech recognition, neural machine translation and sentiment classification. All of these basic applications have been studied with different deep learning techniques [11, 34, 35, 36] and achieved good performance. Correspondingly, many explainable artificial intelligence techniques have been implemented to interpret model structures and relevance between input and output [37, 38, 39], which makes these deep learning models more understandable and trustworthy.

In this work, we interpret the decisions of a deep learning system in sentiment classification task. We use Layer-wise Relevance Propagation (LRP) framework with different strategies to explain the relevance between output label and input words. The relevance of a word is used to indicate to what extent it contributes to an output label. We also systematically evaluate different LRP strategies on linear connections and multiplicative connections in LSTM. Thus, the problems are two-fold:

- How is each input word related to an output label?
- If we use different explaining strategies, what are the impacts they have on the word relevance distribution?

1.4 Objectives and Contributions

One thing we need to notice that the explainable method (i.e., Layer-wise Relevance Propagation) should be implemented on a well-trained model. Otherwise, it is difficult to distinguish whether the bad results come from the model itself or the explainable method. Thus, given the problem statements, the objectives of this work are:

- Train a recurrent neural network with bidirectional LSTM and achieve acceptable performance.
- Explain the relationship between input words and predictions in sentiment classification task.
- Evaluate the influence of different explainable methods on these explanations.

Our contributions are two-fold:

- Extend the LRP framework and propose novel methods to back-propagate relevance through weighted linear connections and multiplicative connections.
- Provide an insight on explaining recurrent neural networks by presenting the performance of the proposed methods in sentiment classification task.

1.5 Thesis Outline

The remaining part of this thesis is organized as follows. In Chapter 2, we demonstrate the basic theories and discuss the work related to the explainable artificial intelligence, especially in natural language processing. Chapter 3 introduces the model structure of sentiment classification task, and different LRP strategies on linear connections and multiplicative connections in LSTM. In Chapter 4, we first present the sentiment model performance. Then we visualize the relevance of each input word to the output label, and evaluate the performance of different LRP strategies.

2. BACKGROUND AND RELATED WORK

In this chapter, we first introduce the background knowledge in deep learning, such as neural network and recurrent neural network. Next, we give an overview of traditional methods and applications in natural language processing. Then we present the current trend in natural language processing using deep learning techniques. Finally, we look into the explainable techniques that are used to interpret the deep neural network.

2.1 Deep Learning

Deep learning is a sub-field of machine learning in artificial intelligence that are inspired by imitating human brain functions in learning data for decision making. According to Dr. Andrew Ng [1], the co-founder of Google Brain, the main purpose of deep learning is two-fold:

- Making learning algorithms more efficient and easier to implement.
- Making revolutionary advances in machine learning and artificial intelligence field.

Given the improvement of computers and the outburst of data volume, it is the right time for deep learning to take off in our life. There are mainly two advantages of deep learning over the traditional machine learning: scalability and hierarchical feature learning.

For the traditional machine learning algorithms, their performance will reach a plateau when the data size is getting larger. As we can see in Figure 2.1, the deep learning algorithms can significantly increase performance with more data. That is to say, if we feed more and more data into deep learning model, its performance keeps getting better. Besides, deep learning allows us to build bigger model, such as larger deep neural networks, for learning from bigger data. Dr. Dean, from Google Brain, has pointed out that the results of neural networks can get better with more

[1] https://www.youtube.com/watch?v=n1ViNeWhC24

data, bigger model and more computations [40].

Another benefit of the deep learning scalability is that deep learning has a strong capability of learning both structured data and unstructured data. Traditionally, machine learning can only process data in tabular formats. For example, we have a list of weather data, each of which contains location, temperature, wind speed and humidity, and then we can train a machine learning model (e.g., random forest) to predict its temperature by inputting location, wind speed and humidity. However, deep learning can process not only data in traditional format but also analogue data, such as images, audios and text documents. Various studies have proved that deep learning algorithms have outperformed machine learning algorithms in both supervised and unsupervised learning [41, 42, 43].

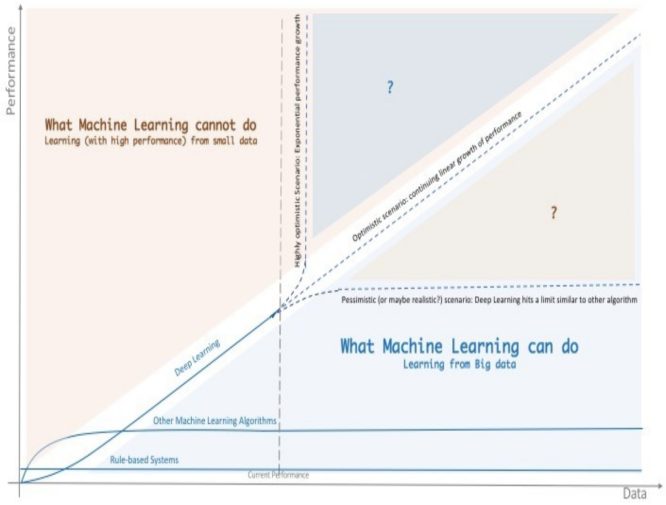

Figure 2.1: Relationship between performance and data quantity for Machine Learning and Deep Learning algorithms, reprinted from [2].

Moreover, deep learning is also capable of performing automatic feature extraction from raw data. In traditional machine learning, we need domain experts to extract features from data manually, since feature engineering can improve the model performance. Deep learning eliminates the human intervention and execute the feature extraction by itself. To be more specific, deep learning

methods can learn data features hierarchically from low-level to high-level [44]. This feature abstraction allows a deep learning system to learn complex mapping from input data to output data, without the need of human-extracted features. For example, when we perform face recognition, deep learning models can automatically extract a person's nose, eyes and cheeks and gradually combine them into new features at higher levels.

Due to these advantages, modern deep learning is getting increasing attentions in both industry and academia. At current stage, the most popular algorithms are Multilayer Perceptron Networks (MLP), Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). In the next two sub-sections, we focus on the neural networks that are most related to this research: MLP and RNN.

2.1.1 Multilayer Perceptron Network

Multilayer Perceptron Network (MLP) is one of the fundamental models in deep learning. It consists of numerous interconnected artificial neurons that pass data among them. These neurons are connected linearly with weights and bias, which are optimized in the network training process, and then transformed through activation functions to outputs.

Neuron is the basic structure of neural network, which works like a neuron in the human brain. A neuron can be seen as a basic processor in the neural network. It receives information from the upper layer, process the information and output to the next layer. One thing we need to point out is that deep learning has nothing to do with biology, although neurons imitate the function of those in a human brain. Instead, it is a mathematical framework for learning representations from data.

Figure 2.2 illustrate how a single neuron works. As we can see, the neuron receives information from three input neurons x_1, x_2 and x_3 through three edges. Next, each of these three neuron values is multiplied by an edge weight plus a bias individually. We then apply an activation func-

tion on the sum of these values as output and pass it to a neuron in the next layer. This rule is generalized in Equation 2.1, where w is the weight vector, b is the bias vector, $f(\cdot)$ is the activation function. x_i, w_i and b_i represent the neuron value, edge weight and bias applied on each input neuron.

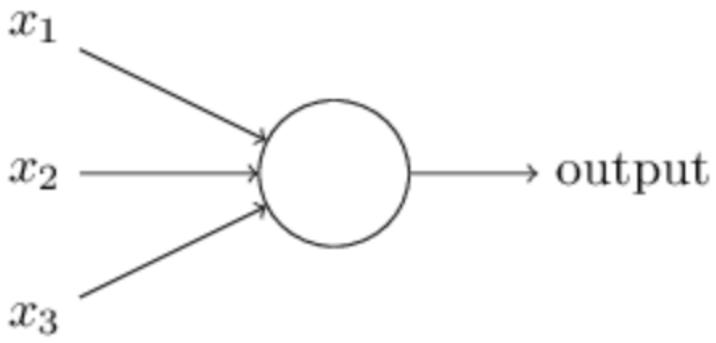

Figure 2.2: A single neuron structure, reprinted from [3].

$$z = f(w \odot x + b) = f(\sum_i w_i \cdot x_i + b_i) \tag{2.1}$$

Different weights represent different importance of features at each layer. When we train a deep learning model, both weights and biases are updated according to the user-defined criterion. The resulting set of these values is able to catch meaningful representations of data at different layers.

The activation function transforms the linear production of inputs to the outputs, which will be passed as the input to the neurons at next layer. Several common activation functions are *Sigmoid* function (Equation 2.2), *Rectified Linear Units (ReLU)* function (Equation 2.3) and *Tanh* function (Equation 2.4). Note that using different activation functions can achieve different effects in deep neural networks. For example, *Sigmoid* function gives a smooth range of input values between 0 and 1, and it is differentiable across its domain. However, *Sigmoid* function is usually not used in intermediate layers due to the vanishing gradient problem. *ReLU* function has a constant deriva-

tive for all inputs greater than zero, which can speed up the training of neural networks. *Tanh* has similar effect as *Sigmoid* function. The difference is that it outputs values between -1 and 1.

$$\sigma(z) = \frac{1}{1 + e^z} \tag{2.2}$$

$$ReLU(z) = max(z, 0) \tag{2.3}$$

$$tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{2.4}$$

In multilayer perceptron networks, there are different numbers of neurons at each layer. A typical MLP structure is presented in Figure 2.3. We can see that each neuron is connected with neurons in neighboring layers by weighted edges.

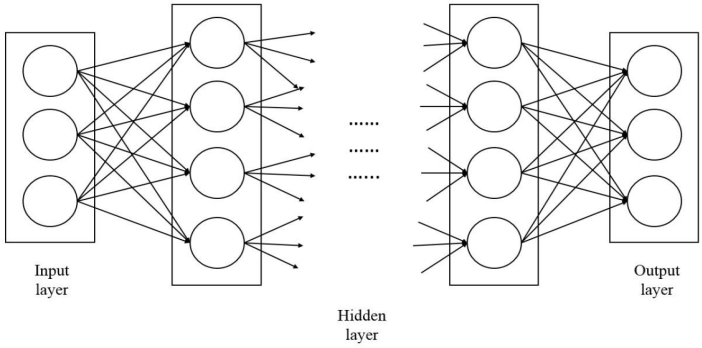

Figure 2.3: Multilayer Perceptron Network.

Generally, layers can be divided into three categories: input layer, output layer and hidden layer. Input layer receives the original data assigned by human, such as image and digits, while output layer presents the final results, such as image recognition outcomes. Hidden layers perform specific tasks on the incoming data and pass the output generated to the next layer. They are trained to extract different levels of data representations as the model goes deeper.

Before training, we need to define a loss function to measure the quality of a candidate solution (i.e., a set of weights and bias). Common loss functions are *Mean Square Error (MSE)* and *Cross-entropy* loss. The *MSE* loss function is used to evaluate the average squared distance between predicted values and true values, while the *Cross-entropy* loss is used in binary-class or multi-class classification problems. These classes are one-hot encoded and the predictions are a set of probability distributions regarding each class. Note that we should choose suitable loss function for each problem in order to train the model correctly.

Given the loss function, the learning problem is then transformed to an optimization problem. The target is to minimize the loss function by updating the weights in neural network. The commonest approach is Gradient Descent. Its main idea is that if a function $F(x)$ is differentiable in the neighboring region of a point u, then $F(x)$ decreases fastest in the direction of its negative gradient at point u. Equation 2.5 gives the general formula of the gradient descent method, where L is loss function, w_{t-1} and w_t are the weights at time step $t-1$ and t, respectively. η is the learning rate which is used to control the step at each optimization iteration. We need to take appropriate step at each iteration in order to reach global minimum point within limited time.

$$w_t = w_{t-1} - \eta \frac{\partial L}{\partial w}|_{w_{t-1}} \tag{2.5}$$

When the training data size gets bigger, it is inefficient to calculate gradient for each training sample. Thus, in order to increase the efficiency, several stochastic gradient descent (SGD) techniques have been developed, such as mini-batch SGD, AdaGrad, RMSProp and Adam. The difference between these SGD techniques lie in the usage of gradient and momentum when updating weights. More details can be found in [45, 46].

2.1.2 Recurrent Neural Network

2.1.2.1 Vanilla RNN

Traditional neural networks flow only in one direction, from input layer to output layer, which means all the inputs are independent with each other. However, when it comes to sequential data (e.g., text and video), there is an issue: the output from previous time step affects the input in the current time step. For example, when we read an article, we understand the content based on our understanding of previous words, and we may also have an idea about what the next paragraph will be. This is because we keep our memory while reading instead of thinking about each word individually.

Recurrent Neural Network (RNN) is used to address this issue. It is not only a feedforward neural network, but also contains connections pointing backward. The left hand side of Fig .2.4 shows a basic recurrent neuron: receiving input x_t, producing an output hidden state h_t and sending h_t back to itself for the next time step. The right hand side of Figure 2.4 presents the vanilla RNN unrolled through time, which illustrates how the RNN loop operation works in sequential order.

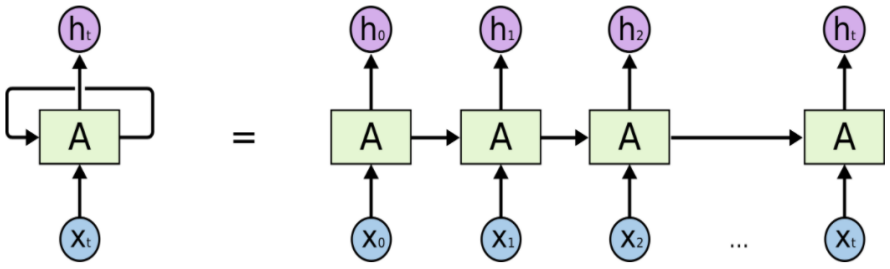

Figure 2.4: Structure of a vanilla RNN, reprinted from [4].

When we train an RNN on long sequences, we need to build a very deep network and run it over many time steps. Similar to any other neural network, it may suffer from the vanishing or exploding gradients problem. The vanishing/exploding gradients problem can be briefly explained as follows: consider the case where an individual gradient of a hidden state with respect to the

previous one is less than one (i.e., $\frac{\partial h_t}{\partial h_{t-1}} < 1$). As we back-propagate across multiple time steps, the product of these gradients get smaller and smaller, leading to the vanishing gradients. On the other hand, if the gradient is larger than one (i.e., $\frac{\partial h_t}{\partial h_{t-1}} > 1$), then the product of gradients get bigger and bigger, leading to exploding gradients.

There are several solutions to alleviate the these problems, such as proper initialization of parameters, non-saturating activation functions, batch normalization, gradient clipping, truncation of sequence length. However, they can neither improve the training speed nor keep the long-term dependency, when we deal with long sequences. In this work, we use Long Short-Term Memory (LSTM) to address these issues.

2.1.2.2 LSTM

Vanilla RNNs are able to consider the recent information when performing tasks at current time step. For example, a language model predicts the next one word based on previous words. Given a sentence "students take classes in", it is obvious that the next word is "school", and we notice that we don't need more context in the previous sentence. In this case, vanilla RNNs can handle this problem, since the gap between the relevant context and the prediction spot is small. However, when the gap is getting larger, there is a problem that RNN may not be able to connect the relevant information far from the prediction spot.

LSTM is used to address this issue. It is a variant of RNN that is able to detect long-term dependencies in the sequential data [47]. Basically, LSTM implements recurrence with four fully-connected layers interacting in a way such that the proper part of long-term and short-term memory can be kept when performing current task. Figure 2.5 presents the structure of a typical LSTM cell. It consists of two states (i.e., short-term state h_t and long-term state c_t), and three gates (i.e., forget gate, input gate and output gate).

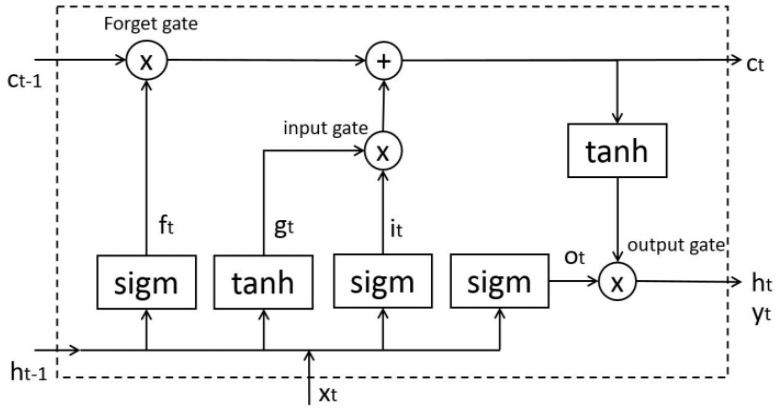

Figure 2.5: Structure of a LSTM cell.

Forget gate is used to control what information should pass. The information is collected from the input x_t at current step and hidden state h_{t-1} from previous step. Equation 2.6 is the formula of forget gate. We notice that there is a sigmoid function that are used to make this decision. The closer the output f_t is to zero, the more information we should forget.

$$f_t = \sigma(W_{xf} \cdot x_t + W_{hf} \cdot h_{t-1} + b_f) \tag{2.6}$$

Input gate is used to update the cell state c_t. Equations 2.7 present the flow of input gate. First, we pass the current input x_t and previous hidden state h_{t-1} to a *tanh* function for regulation. Then we multiply g_t with another gate value i_t to decide what information we should keep. After that, we combine this retained information with the previous cell state c_{t-1} to get the current cell state c_t.

$$\begin{aligned} i_t &= \sigma(W_{xi} \cdot x_t + W_{hi} \cdot h_{t-1} + b_i) \\ g_t &= tanh(W_{xg} \cdot x_t + W_{hg} \cdot h_{t-1} + b_g) \\ c_t &= f_t \odot c_{t-1} + i_t \odot g_t \end{aligned} \tag{2.7}$$

Output gate is used to decide the hidden state h_t at the current step. It uses a gate neuron o_t to decide how much information h_t should keep according to the current cell state c_t. The new hidden state and new cell state is then passed to the next step.

$$
\begin{aligned}
o_t &= \sigma(W_{xo} \cdot x_t + W_{ho} \cdot h_{t-1} + b_o) \\
h_t &= o_t \cdot tanh(c_t)
\end{aligned}
\tag{2.8}
$$

Note that in equations 2.6, 2.7 and 2.8, W_{x*}, W_{h*} are weights connected to current input x_t and hidden state h_{t-1} at previous step. b_* is the bias term.

2.1.2.3 Bi-directional LSTM

Standard LSTM is uni-directional. This means that it always reads the sequential data from left to right. For example, given a sentence "I am a student", the uni-directional LSTM receives input in the following order: "I", "am", "a", "student". However, future words may also have influence on predicting the current word. Bi-directional LSTM is able to include features from both past and future at current step. It has been proved that Bi-LSTM has outperformed the state-of-the-art models in many natural language processing tasks, such as speech recognition [48] and sequence tagging [49]. Figure 2.6 presents the structure of a bi-directional RNN, where each neuron represents an LSTM cell. The network consists of a forward LSTM and a backward LSTM, which receive the input from two directions. Then they generate a set of forward hidden states ($\overrightarrow{h_t}$) and backward hidden states ($\overleftarrow{h_t}$), concatenate ($[\overrightarrow{h_t}, \overleftarrow{h_t}]$) and pass them to the next step.

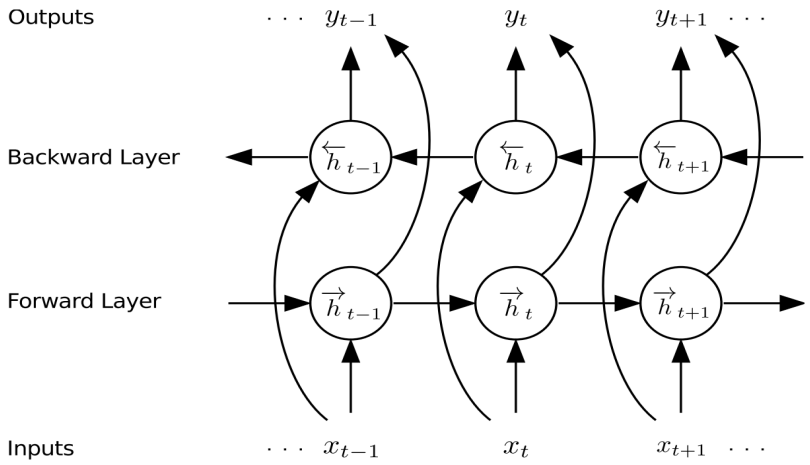

Figure 2.6: Structure of a bi-directional RNN, reprinted from [5].

2.2 Natural Language Processing

In this section, we introduce the conventional techniques in Natural Language Processing (NLP) that are related to our work. These techniques include Bag of Words, word embedding and their applications in sentiment analysis.

2.2.1 Bag of Words

The Bag-of-Words (BoW) model is a simple representation that is commonly used in natural language processing and information retrieval area. It collects the frequency of each word within a sentence or document, so that each document can be described as a vector. The elements in this vector represent the occurrence of each word in the document. In this model, we disregard the information about semantics, syntax and word orders, and just use word occurrence as the text feature. One advantage of BoW is that it is easy to implement and compare documents if they have similar content. Previous studies have proved its effectiveness in many NLP fields, such as document categorization [50], machine translation [51] and fraud detection [52].

However, when we have documents with huge vocabularies, each vector becomes highly sparse. This could cause memory storage issue and increase the complexity of further operations. Another issue is that BoW may contain words with high frequency but useless information, such as "the". Thus, the way to solve this word frequency issue is *TF-IDF* [53], where TF stands for term frequency and IDF stands for inverse document frequency. In doing this, words with high document frequency is penalized, while meaningful words are benefited with more weights.

2.2.2 Word Embedding

Word embedding is a learned representation that maps words to vectors consisting of real numbers. It includes a suite of techniques where each word is represented as a sparse or dense vector space. The basic form of word embedding is *One-hot encoding*. It uses a large sparse binary vector

to represent a word, where all of its elements are zero except one position k (k is the position of this word in vocabulary). Although the one-hot encoding approach is easy to use and understand, it still suffers from the curse of dimensionality once we have a huge vocabulary. Moreover, it is not able to cluster the words with similar meanings and include the sequential information in sentences.

One of the advanced word embedding techniques that preserve semantic relationships between words is *Co-occurrence Matrix*. It counts the frequency of two or more words that occur together in documents. More specifically, it calculates $count(w_{next}|w_{current})$, where $w_{current}$ and w_{next} are the current word and its next neighbor. Thus, we can repeat this process for each word in vocabulary and the resulting matrix is the co-occurrence matrix, where each row represents a word vector. Although it provides more accurate word representations, it still fights with the curse of dimensionality, since each word vector has a huge dimension. The idea to solve this issue is using neural networks to represent words in denser vectors.

2.2.2.1 Word2Vec

One of the most popular neural network based word embedding is *Word2Vec* [54, 55]. *Word2Vec* is able to map each unique word in vocabulary to a low-dimension dense vector space (typically hundreds of dimensions). It trains shallow neural networks, i.e., Continuous Bag-of-Words model (CBOW) and Continuous Skip-Gram model, to construct the linguistic context of words. Both models can preserve the semantic and syntactic relationships by predicting a word with its nearby words. The difference is that CBOW predicts word by its context while the skip-gram model predicts its context by the target word. The resulting word vectors can cluster the words that share common contexts in the corpus. One of the interesting characteristics of this word embedding method is that it can capture word similarities, such that similar words can be found by vector arithmetics, e.g., *king - man + woman = queen*. Due to this property, *Word2Vec* has now been implemented in many researches, such as text classification [56], named entity recognition [57] and sentiment analysis [58].

2.2.2.2 GloVe

Another well-known embedding technique is *GloVe* [59]. Unlike *Word2Vec*, *GloVe* uses co-occurrence statistics to relate similar words while distancing unrelated words. Thus, *GloVe* is basically a count-based model, where we first construct a big co-occurrence matrix and then perform dimensionality reduction on it. After factorization, each row vector in the resulting lower-dimensional matrix is used to represent each word in vocabulary. This process is achieved by training a log-bilinear regression with an objective of minimizing a reconstruction loss. It is known that *GloVe* combines the benefits of global statistics using matrix factorization and local statistics using context-window methods.

The core idea of *GloVe* is to decide the relative relation between three words by using the ratio of co-occurrence probabilities. For example, there are three words w_i, w_j and w_k. Now suppose w_k appears more frequently in the context of w_i than w_j. Thus, the ratio of co-occurrence $\frac{P(w_k|w_i)}{P(w_k|w_j)}$ should be large. If w_k appears more frequently in the context of w_j than w_i, the co-occurrence ratio $\frac{P(w_k|w_i)}{P(w_k|w_j)}$ should be small. If w_k appears frequently both in the context of w_j and w_i, or it appears rarely in both context, then the co-occurrence ratio should be close to one. In doing this, we are able to differentiate the relevant words with irrelevant words. The resulting word representations show an interesting linear property in the vector space, such that the difference between vectors can capture the meaning specified by parallel of two words [59].

In addition, *GloVe* is easier to train with more data due to its parallelization. Therefore, given all these advantages, *GloVe* has obtained the most attention in deep learning, especially in the recurrent neural network field. Such researches as name entity recognition [60], emotion classification [61] and speech detection [62] have all reached improved performance with *GloVe*.

In this work, we use pre-trained *GloVe* as the initialization of the word embedding layer in our Bi-LSTM sentiment analysis model.

2.2.3 Sentiment Analysis

Sentiment analysis is one of the fundamental tasks in natural language processing. The objective is to identify and extract the subjective information from texts. Typically, in a sentiment analysis task, we classify the expressed opinions or affective states from a given document or sentence as positive, negative or neutral. Several examples are given below:

- Input: It is one of the greatest family - oriented, fantasy - adventure movies ever.
  Output: Positive.
- Input: If you sometimes like to go to the movies to have fun, wasabi is a good place to start.
  Output: Positive.
- Input: It becomes gimmicky instead of compelling.
  Output: Negative.

Moreover, there are also other type of affective states, such as emotion (i.e., joyful, sad, angry, fearful), mood (i.e., cheerful, gloomy, irritable, depressed) and personality traits (i.e., nervous, anxious, hostile, humble) [63].

Sentiment analysis is widely applied in customer reviews, survey responses and many other data in healthcare, social media and marketing. Given the outburst of data volumes, sentiment analysis is really helpful to process different types of data at scale, identify issues in real-time and create consistent criteria. It has also been proven to be a valuable technique in recommender systems that recommend products to users based on their perferrence [64, 65, 66].

Sentiment analysis is one of the most challenging tasks due to its inability to perform well in every domain and incapability to handle complex sentence structure (e.g., double negation). Existing methods to sentiment analysis can be divided into categories: knowledge-based approaches, statistical approaches and hybrid approaches.

Knowledge-based approaches define a set of rules that are collected from domain-based facts [67, 68]. Then we use these facts to perform sentiment analysis by inference. The advantage of these approaches is that it can provide explanation to its local thinking process, which makes the system more reliable. However, collecting these facts itself can be cumbersome.

On the other hand, statistical approaches implement machine learning techniques. Typical methods are Naive Bayes [69], logistic regression [70], support-vector machine (SVM) [71] and K-Nearest Neighbors [72]. Although these methods can reach higher accuracy than traditional knowledge-based approaches, we need to manually extract features, such as word frequency, in order to achieve reliable results. With the arrival of deep learning, many researchers start to focus on the implementation of neural network on sentiment analysis. Convolutional neural networks (CNN) and recurrent neural networks (RNN) are the two well-known techniques [36]. Liao et al. [73] designed a simple convolutional neural network model to extract sentiment information from massive Twitter texts, which outperformed conventional SVM and Naive Bayes methods. Li et al. [74] used LSTM to classify emotion of sentences, and they proved that the LSTM can achieve better performance than the vanilla RNN. Wang et al. [75] combined CNN and RNN for sentiment classification for short texts, and the results are better than the state-of-the-art on benchmark datasets.

Hybrid approaches join both the machine methods and knowledge-based methods. They first extract lexicon features from raw documents and then feed these features to machine learning models [76, 77, 78].

In this work, we build the recurrent neural network based on Bi-LSTM. Due to the lack of explainability in deep learning models, we use explainable methods to interpret the relevance between its decision and input words.

2.3 Explainable Artificial Intelligence

In this section, we introduce Explainable Artificial Intelligence (XAI). First, we give an overview about what XAI is and its contributions. Second, we discuss about the understandable models in conventional machine learning field. Then, we introduce different types of explainable approaches based on local regions and global behaviors. Lastly, we discuss the related works in XAI that have been adapted in natural language processing.

2.3.1 Explainability of Artificial Intelligence

Traditionally, when people design an expert system, they mainly focus on defining a set of complicated rules. These rules, which are most commonly used as knowledge representation, are usually defined in an "if-then" structure [79]. For example, a doctor may make the right decision on diagnosis through an expert system, given a set of typical symptoms. As we can see, these rules can be easily understood by human. Thus, the entire expert system is also easy to explain. However, modern systems based on artificial intelligence are totally different with the conventional rule-based systems. The system rules are now defined in terms of a huge number of parameters that are trained through many iterations. Although the performance of these systems has been largely improved, it is difficult for us to understand the reason behind model decisions.

For the time being, there is still no consensus on understanding what the explainability is, but many studies on explainable models and techniques have enriched the meaning of explainability. Gunning et al. [14] define XAI as a set of machine learning techniques that can let human understand and trust AI systems. Doran et al. [80] give another concept of explainability - an AI system that can automatically reason its decision without any post intervention by human. However, whether the explanations are understandable by human is totally dependent on the audience themselves. Thus, Arrieta et al. [81] propose that given an audience, XAI is able to justify model's decision by clarifying its functioning and making it easy to understand. Generally, it is easy for us to un-

derstand simple models, such as linear classifier (especially in two dimension), while complicated models, such as CNN and RNN, are difficult to understand because of their nonlinear structures and optimization iterations. In this study, we regard the explainability of a deep learning system as the ability to explain the reason behind its decision that is acceptable and understandable by human.

With the arrival of deep learning and big data era, deep neural network is becoming a common technique to build AI systems. Thus, it is in great need that we should explain these models in a human-understandable way. The main contributions of XAI are listed as follows [17]:

- **Trustworthiness:** Modern AI systems are treated as black boxes. That is, people can only interact with the inputs and outputs, but they cannot check what is happening during the decision making process. In some area such as medical diagnosis, it is important for researchers to justify the decisions of the system, since the model does not understand human symptoms. Thus, XAI can increase the mutual trust between human and systems.

- **Model improvement:** In most studies, people usually directly implement existing deep learning models to their applications, such as speech recognition and machine translation. Nevertheless, this methodology may reach a performance plateau due to the limitations of neural networks. Through explaining these models, we can modify their structures and achieve better results.

- **New insights acquisition:** Since deep neural networks can learn rules that human cannot understand. They may generate new insights to real events in our life. For example, AI systems outperform human chess players, so they may have applied new strategies that are previously not known to human. In this sense, we can extract these special strategies from XAI and apply them in games between human.

- **Legal compliance:** XAI also has a great impact on legal issues, since the black box models now cannot justify the reason to problems about human rights. One example is whether a

criminal is allowed to be paroled. With XAI, we can provide the justification that is accepted by the public.

2.3.2 Interpretable Machine Learning Models

A model is regarded as interpretable if it is understandable without human intervention. Many existing machine learning models are interpretable models. For example, logistic regression model assumes a linear relationship between predictors and target. Many studies have used different techniques to analyze the soundness of logistic regression model [82, 83, 84].

Another example is decision tree, which has a flowchart structure with nodes and branches. Thus, it is straightforward for people to follow from root to leaf nodes and explain whether the final decision is reasonable. However, when we transform decision trees to decision forests, it may increase the complexity if we want to explain them. Sagi et al. [85] used a set of rule conjunctions to reduce the original decision forest into a single interpretable decision tree. It has been proved that the reduced model can preserve the predictive capability while providing efficient explanations to human.

Other machine learning models, such as K-Nearest Neighbors and bayesian models, are also easy to interpret due to their structures and rationales [86, 87, 88].

2.3.3 Local Explainable Approaches

Given a well-trained model, the local explainable approaches aim to answer the question: why does the model make a certain decision for an input instance? For example, if there is a model trained for image classification, we want to know how each pixel of an image in the test set is related to the predicted label. Thus, in local explainable methods, each input and prediction pair is studied individually to explain how a model behaves. There are two types of local explainable methods, i.e., model-aware methods and model-unaware methods, depending on whether an ex-

plainable method uses the parameters directly from the model.

Model-aware methods utilize the parameters directly from a model. For convolutional neural networks, many previous works focused on CNN visualization by highlighting relevant pixels that have large impact on the final predictions [22, 89, 90]. Although these methods can provide high-quality visualization, they are not class-discriminative, which means that the generated images may be very similar with respect to different classes. Another strategy is to detect objects from images based on their class labels [91, 92, 93]. Zhou et al. [23] proposed to use class activation maps (CAM) to highlight the objects detected by CNNs. However, this technique only applies to CNN architecture with average-pooling before prediction. Selvaraju et al. [94] extended CAM to Grad-CAM using gradient signal, so that it can be adapted to any CNN architecture. Bach et al. [28] proposed the Layer-wise Relevance Propagation (LRP) technique to back-propagate the output score to the input instance. This method can calculate contribution of each pixel in each input image. In recurrent neural networks area, one way to explain RNN is visualizing hidden nodes. Strobelt et al. [95] developed an interactive tool to analyze the hidden states pattern given a pre-defined input range. This technique can also find similar patterns in large data set. Another way to explain RNNs is analyzing the contribution of each input word to predictions. Arras et al. [24] implemented Layer-wise Relevance Propagation method on LSTM and highlighted the words that have positive and negative impact, respectively.

Model-unaware methods derive explanations mainly based on sensitivity. The sensitivity can be computed as gradients used in back-propagation. Simonyan et al. [89] calculated partial derivatives of class score with respect to an input image, and pointed out the most sensitive part of image. The drawback of this approach is that we cannot distinguish whether the pixels have positive or negative effect on the output label. Li et al. [96] extended this approach in natural language processing. They highlighted the input neurons that have big contribution to the final classification. Another approach is called Contextual Decomposition (CD) [97]. It separates the forward pass of

LSTM into two parts (relevant part and irrelevant part), and combines these two hidden states to give a relevance score of each word. The drawback is that it results high computation cost, since it requires multiple forward passes.

2.3.4 Global Explainable Approaches

The global explainable approaches aim to answer the question: how does the model make predictions over the entire input space? Similar to the local explainable approaches, global explainable approaches can be categorized into model-aware methods and model-unaware methods.

For model-aware methods, one way to explain neural network is analyzing the representation of each neuron or each layer of neurons. Bau et al. [98] proposed a network dissection strategy to measure what each hidden node represent in CNN. Consistent with the hypothesis, the results proved that different layers in CNN represent partial features for images. Karpathy et al. [25] visualized the functions of different cells inside LSTMs and showed that these cells can handle different features, such as line length and quotes, when processing documents.

For model-unaware methods, there is few research about explaining models in a global sense. Ribeiro et al. [99] proposed a novel technique to give global explanation of any model by selecting a subset of representative examples. Although this method is flexible for many models, such as random forest and neural network, it is not efficient when dealing with larger datasets.

2.3.5 Explainable Approach in Our Work

In this work, we use Layer-wise Relevance Propagation (LRP) technique to explain LSTMs. LRP is previously used in CNN [28] and then extended to RNN [24]. The difference is that the word vectors are the basic units in RNN instead of single pixels. The main idea of this approach is quantitively visualizing the relevance of each neuron at the input layer to the final decision at

output layer.

The LRP approach is significantly different from the attention mechanism [33]. The attention scores are attached separately to LSTM hidden states at each time step, but not the input words directly. This means the attention mechanism can only explain the model in local scope. In contrast, relevance can be used to visualize the relationship between any two neurons in the neural networks. Another advantage of LRP over gradient-based methods is that it is invariant against the nonlinear activations, which saves lots of computational efforts.

However, the LRP framework has not been extensively studied in RNN. In this work, we use different methods on weighted linear connections and multiplicative connections in LSTM, and evaluate them thoroughly in sentiment analysis. The results may provide insights on how to improve LRP in the future.

3. METHODOLOGY

In this chapter, we first introduce the Bi-LSTM-based recurrent neural network for sentiment classification, including its architecture and detailed parameters. Next, we introduce the details of general Layer-wise Relevance Propagation framework as well as different methods we use on weighted linear connections and multiplicative connections in LSTM.

3.1 Bi-LSTM Model for Sentiment Classification

In this sentiment classification task, we use recurrent neural network based on bi-directional LSTM. Figure 3.1 illustrates the model architecture.

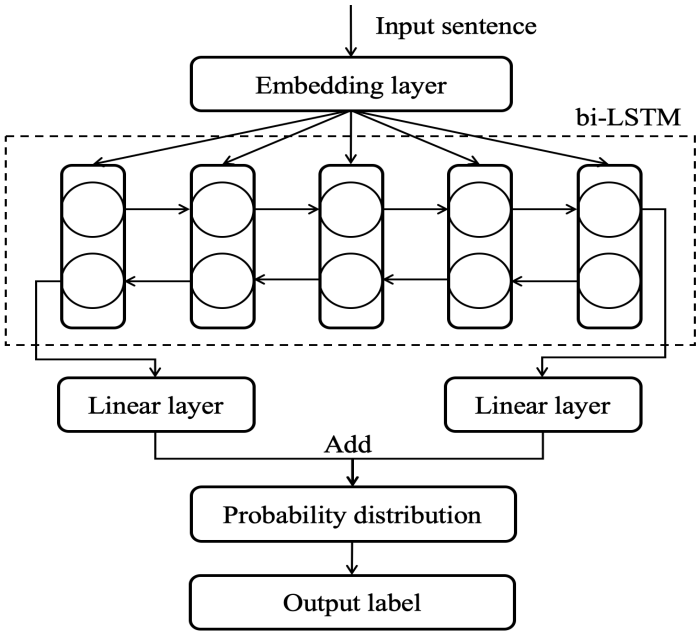

Figure 3.1: The architecture of sentiment classification model.

Given the input sentence, the encoder first embeds each word to a real-valued vector through an embedding layer, which is initialized with the pre-trained word embedding *GloVe* [59]. Then the recurrent neural network with Bi-LSTM reads the input sentence in the forward and backward

direction, respectively. For example, suppose we have an input sentence - "this is a fascinating movie". At the first time step ($t = 1$), the forward encoder reads the word "this". At $t = 2$, it reads the word "is" while keeping the information of word "this". The forward encoder keeps reading until the end of this sentence. Similarly, the backward encoder reads the sentence in the reverse order: "movie fascinating a is this". In doing this, the model captures the information from both past words and future words at each time step.

As a result, we obtain a set of forward hidden states ($\overrightarrow{h_t} = f(\overrightarrow{h}_{t-1}, x_t)$) and backward hidden states ($\overleftarrow{h_t} = f(\overleftarrow{h}_{t-1}, x_t)$), where $f(\cdot)$ represent the Bi-LSTM encoder. In the next step, we use the forward hidden states and backward hidden states at the last time step, each of which go through a linear layer respectively. Finally, we add the resulting two states together to get the probability distribution of each label. We predict the sentiment label with the largest probability value.

3.2 Layer-wise Relevance Propagation for Bi-LSTM

The Layer-wise Relevance Propagation (LRP) was first introduced by Bach et al. [28] on convolutional neural networks. Here we extend this framework on LSTM and compute neuron-level relevance from output layer to input layer. We use a simple three-layer neural network shown in Figure 3.2 to illustrate how the LRP works.

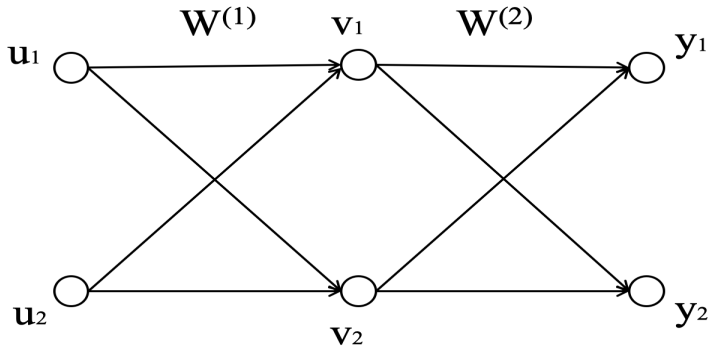

Figure 3.2: A simple three-layer neural network for illustrating LRP.

As we discussed in Chapter 2, the LRP redistributes the relevance from the neurons of output layer back-propagate to the neurons of input layer. We notice in Figure 3.2 that there are two neurons in each layer. Starting from the left, u_1, u_2 are input neurons, v_1, v_2 are hidden neurons and y_1, y_2 are output neurons. In this example, we use $R_{i\leftarrow j}$ to represent the relevance message from neuron j to neuron i. R_i is the relevance score of neuron i. $W_{ij}^{(m)}$ is the weight connection between neuron i and neuron j of different layers. For example, $W_{12}^{(1)}$ is the weight between neuron u_1 and neuron v_2. $R_{v_1\leftarrow y_2}$ is the relevance message passing from neuron y_2 to neuron v_1. R_{u1} is the total relevance received by neuron u_1. In the following, we present how LRP pass relevance from each neuron of output layer to input neurons.

First, LRP redistribute the relevance from output neurons to hidden neurons. In the example case, it sends a message from neurons y_1, y_2 to neurons v_1, v_2. We can calculate the relevance message from y_1 to v_1 and v_2 as follows:

$$\begin{aligned} R_{v_1\leftarrow y_1} &= \frac{W_{11}^{(2)} \cdot v_1}{W_{11}^{(2)} \cdot v_1 + W_{21}^{(2)} \cdot v_2} R_{y_1} \\ R_{v_2\leftarrow y_1} &= \frac{W_{21}^{(2)} \cdot v_2}{W_{11}^{(2)} \cdot v_1 + W_{21}^{(2)} \cdot v_2} R_{y_1} \end{aligned} \tag{3.1}$$

where R_{y_1} is equal to the neuron value of y_1 since it is an output neuron. We also notice that the activation functions are ignored. This is because LRP is invariant with the non-linear equations [28]. Similarly, we can calculate the relevance message from y_2 to v_1 and v_2 as follows:

$$\begin{aligned} R_{v_1\leftarrow y_2} &= \frac{W_{12}^{(2)} \cdot v_1}{W_{12}^{(2)} \cdot v_1 + W_{22}^{(2)} \cdot v_2} R_{y_2} \\ R_{v_2\leftarrow y_2} &= \frac{W_{22}^{(2)} \cdot v_2}{W_{12}^{(2)} \cdot v_1 + W_{22}^{(2)} \cdot v_2} R_{y_2} \end{aligned} \tag{3.2}$$

Next, the relevance can be further back-propagated to the input neurons. For example, $y_1 \to v_1 \to u_1$, $y_1 \to v_2 \to u_1$ (Equation 3.3) and $y_1 \to v_1 \to u_2$, $y_1 \to v_2 \to u_2$ (Equation 3.4).

$$\begin{aligned} R_{u_1 \leftarrow y_1} = & \frac{W_{11}^{(1)} \cdot u_1}{W_{11}^{(1)} \cdot u_1 + W_{21}^{(1)} \cdot u_2} R_{v_1 \leftarrow y_1} \\ & + \frac{W_{12}^{(1)} \cdot u_1}{W_{12}^{(1)} \cdot u_1 + W_{22}^{(1)} \cdot u_2} R_{v_2 \leftarrow y_1} \end{aligned} \tag{3.3}$$

$$\begin{aligned} R_{u_2 \leftarrow y_1} = & \frac{W_{21}^{(1)} \cdot u_2}{W_{11}^{(1)} \cdot u_1 + W_{21}^{(1)} \cdot u_2} R_{v_1 \leftarrow y_1} \\ & + \frac{W_{22}^{(1)} \cdot u_2}{W_{12}^{(1)} \cdot u_1 + W_{22}^{(1)} \cdot u_2} R_{v_2 \leftarrow y_1} \end{aligned} \tag{3.4}$$

We can follow the similar procedure to calculate $R_{u_1 \leftarrow y_2}$ and $R_{u_2 \leftarrow y_2}$.

Lastly, we collect all the relevance messages for each neuron and obtain its total relevance received. Take neuron u_1 for an example:

$$\begin{aligned} R_{u_1} = & R_{u_1 \leftarrow y_1} + R_{u_1 \leftarrow y_2} \\ & + \frac{W_{11}^{(1)} \cdot u_1}{W_{11}^{(1)} \cdot u_1 + W_{21}^{(1)} \cdot u_2} R_{v_1} \\ & + \frac{W_{12}^{(1)} \cdot u_1}{W_{12}^{(1)} \cdot u_1 + W_{22}^{(1)} \cdot u_2} R_{v_2} \end{aligned} \tag{3.5}$$

where $R_{v_1} = R_{v_1 \leftarrow y_1} + R_{v_1 \leftarrow y_2}$ and $R_{v_2} = R_{v_2 \leftarrow y_1} + R_{v_2 \leftarrow y_2}$.

We can find out that the total relevance in each layer satisfies the conservation property. In this case,

$$R_{u_1} + R_{u_2} = R_{v_1} + R_{v_2} = R_{y_1} + R_{y_2} \tag{3.6}$$

Therefore, we can define the general rule of LRP as follows. Suppose R_j is the relevance received by neuron j in layer $l+1$ and R_i is the relevance received by neuron i in layer l. Then the

relationship between R_i and R_j is:

$$R_i = \sum_j R_{i \leftarrow j} = \sum_j \frac{z_i \cdot W_{ij}}{\sum_i z_i \cdot W_{ij}} R_j \tag{3.7}$$

where z_i is the value of neuron i, W_{ij} is the weight between neuron i and neuron j.

As we notice in the Bi-LSTM model (Figure 3.1), there are a set of neurons at the output layer, each of which stands for the probability of a class label, such as positive or negative. In this work, we only keep the output neuron with highest value and mask out the rest (i.e., setting to zero). In doing this, we can visualize the relevance of each input word to the predicted label.

According to the LSTM formulas (equations 2.6, 2.7, 2.8), there are two types of connections: multiplicative connections and weighted linear connections. In this work, we use different strategies for each type of the connections and evaluate their impacts on the generated explanations.

3.2.1 Multiplicative Connection

For the multiplicative connection, it has a typical form:

$$z_j = z_g \odot z_s \tag{3.8}$$

where z_j is the output neuron, z_g the gate neuron and z_s is the source neuron. In this work, we use three different strategies to distribute the relevance of neuron z_j to gate neuron and source neuron: relative distribution, absolute distribution and zero-one distribution.

3.2.1.1 Relative distribution

For the relative distribution, we pass the relevance of output neuron based on the relative values of gate neuron and source neuron. This strategy was first proposed by [30] and implemented in

explaining a neural machine translation application. It has a typical form:

$$
\begin{aligned}
R_g &= \frac{z_g}{z_g + z_s + \epsilon \cdot sign(z_g + z_s)} R_j \\
R_s &= \frac{z_s}{z_g + z_s + \epsilon \cdot sign(z_g + z_s)} R_j
\end{aligned}
\tag{3.9}
$$

where R_j is the relevance of output neuron j, R_g and R_s are the relevance of gate neuron and source neuron, respectively. We notice that there is a stabilizer ϵ added on the denominator. Typically, ϵ is a small positive number (i.e., 0.001) avoiding zero division. However, this term can cause numerical instability [38]. In this work, we eliminate this stabilizer by applying a sigmoid function on both gate neuron and source neuron (Equation 3.10). Recall that the gate neuron is the output of a sigmoid function. Thus, by using this strategy, the denominator can never reach a value that is extremely close to zero.

$$
\begin{aligned}
R_g &= \frac{\sigma(z_g)}{\sigma(z_g) + \sigma(z_s)} R_j \\
R_s &= \frac{\sigma(z_s)}{\sigma(z_g) + \sigma(z_s)} R_j
\end{aligned}
\tag{3.10}
$$

3.2.1.2 Absolute distribution

The absolute distribution is similar to the relative distribution. The difference is that we pass the relevance of output neuron based on the absolute values of gate neuron and source neuron. In this work, we also apply a sigmoid function on both of them. The typical form is as follows:

$$
\begin{aligned}
R_g &= \frac{\sigma(|z_g|)}{\sigma(|z_g|) + \sigma(|z_s|)} R_j \\
R_s &= \frac{\sigma(|z_s|)}{\sigma(|z_g|) + \sigma(|z_s|)} R_j
\end{aligned}
\tag{3.11}
$$

3.2.1.3 *Zero-one distribution*

The zero-one distribution was first proposed by [24]. It passes all the relevance of the output neuron to the source neuron (Equation 3.12). The intuition behind this method is that the gate neuron is used to determine the fraction of information should contribute to the lower layer. Thus, the gate contribution has already been included when we pass the relevance flow to the source neuron.

$$\begin{aligned} R_g &= 0 \\ R_s &= R_j \end{aligned} \tag{3.12}$$

3.2.2 Weighted Linear Connection

The weighted linear connection has a typical form:

$$z_j = \sum_i z_i \cdot W_{ij} + b_j \tag{3.13}$$

where z_i is the lower-layer neuron, z_j is an upper-layer neuron, w_{ij} and b_j are the weight and bias, respectively. In this work, we use three different strategies to distribute the relevance of upper-layer neuron z_j to lower-layer neuron z_i: $\epsilon-$ rule, $\beta-$ rule and $abs-$ rule.

3.2.2.1 $\epsilon-$ *rule*

The $\epsilon-$ rule is most frequently used in CNN and RNN [24, 39, 100]. It follows the general rule (Equation 3.7) and add a stabilizer on the denominator (Equation 3.14):

$$R_{i \leftarrow j} = \frac{z_i \cdot W_{ij} + \delta \cdot (\epsilon \cdot sign(z_j) + b_j)/N}{z_j + \epsilon \cdot sign(z_j)} R_j \tag{3.14}$$

where $z_j = \sum_i z_i \cdot W_{ij}$, N is the number of neurons at lower layer that connect to z_j, δ is a bias factor to determine whether bias should be considered, ϵ is a small stabilizer that prevents zero division. In this work, we set $\epsilon = 0.001$. The drawback of this method is that it does not perfectly

satisfy the conservation property due to the stabilizer ϵ, and the stabilizer can cause numerical instability [101].

3.2.2.2 $\beta-$ rule

The $\beta-$ rule was first initialized by [28] and implemented on CNN, but there is little research studying this method on RNN. The main idea of this method is handling positive connections (activator) and negative connections (inhibitor) separately. More specifically, for each weighted connection z_{ij}, it passes the relevance as follows:

$$R_{i \leftarrow j} = ((1+\beta) \cdot \frac{z_{ij}^{+}}{z_{j}^{+}} - \beta \cdot \frac{z_{ij}^{-}}{z_{j}^{-}}) R_j \tag{3.15}$$

where $z_{ij}^{+} = max(0, z_{ij})$, $z_{ij}^{-} = min(0, z_{ij})$. β is a pre-defined parameter ($\beta \geq 0$). There are two drawbacks:

- It does not satisfy the conversation property for any β value. For example, suppose we have a fully connected layer with all three positive connections. If we set β to be 1, we actually double the relevance from the neuron of upper layer.
- To the best of author's knowledge, it is still an unknown whether there is an optimal β that produces the best performance of LRP in RNN.

In this work, we propose a new strategy to assign β value based on connections. In other words, we use different β values to different sets of connections (Equation 3.16).

$$\beta = \frac{count_{neg}}{max(count_{pos} + count_{neg}, 1)} \tag{3.16}$$

where $count_{pos}$ and $count_{neg}$ are the number of positive connections and negative connections, respectively. We use a max function in the denominator to avoid zero division.

3.2.2.3 $abs-$ rule

An alternative method we propose in this work is passing the relevance based on absolute connections (Equation 3.17). In this way, we pass positive message to lower-layer neurons only if the relevance of upper-layer neuron is positive. Otherwise, we pass negative message.

$$R_{i \leftarrow j} = \frac{|z_{ij}|}{|z_j| + \epsilon} R_j \tag{3.17}$$

3.2.3 Method Naming Convention

In this work, we combine different strategies on multiplicative connections and weighted linear connections. We name our combined methods in the following format:

[linear method] - LRP - [multiplicative method].

For example, "$\epsilon - LRP - relative$" represents using ϵ-rule on linear connections and $relative$ distribution on multiplicative connections, while "$\beta - LRP - abs$" represents using β-rule on linear connections and $absolute$ distribution on multiplicative connections. Moreover, we label the zero-one distribution as "all". Thus, "$\epsilon - LRP - all$" represents using ϵ-rule on linear connections and $zero - one$ distribution on multiplicative connections.

3.3 Sensitivity Analysis

As a reference, we implement gradient-based sensitivity analysis in this work. Similar to LRP, the sensitivity analysis method produces a relevance distribution for each word in a sentence [37, 89]. The difference is that these relevances are obtained by squared partial derivatives (Equation 3.18):

$$R_i = \left(\frac{\partial f(x)}{\partial x_i} \right)^2 \tag{3.18}$$

4. RESULTS AND DISCUSSION

In this chapter, we present the results in our work. First, we introduce the dataset for training and test, and present the performance of Bi-LSTM model. Next, we decompose the sentiment to input words and visualize the words that have positive relevance and negative relevance. Then we present the representative words for each class label using different methods on multiplicative connections and linear connections. In addition, we perform word perturbation experiments to validate the rationality of relevance distribution among these methods. Lastly, we focus on single classification cases by adding negations or double negations, in order to explain the model behavior in different semantic environment.

4.1 Bi-LSTM Model Performance

4.1.1 Data

We use the Stanford Sentiment Treebank (SST) dataset in this sentiment classification task. It contains 8,544 sentences in the training set, 1,101 sentences in the validation set and 2,210 sentences in the test set. The class labels are positive, negative and neutral. The following sentences are two examples in the training set:

- "Positive": the rock is destined to be the 21st century's new "conan" and that he's going to make a splash even greater than arnold schwarzenegger, jean - claud van damme or steven segal.
- "Negative": if the tuxedo actually were a suit , it would fit chan like a $99 bargain - basement special.

4.1.2 Performance

In this three-class sentiment classification task, we build the model using bi-directional LSTM. It consists of one embedding layer, one hidden layer of bi-LSTM and two linear layers, as shown in

Figure 3.1. The embedding layer is initialized with 300-dimensional pre-trained GloVe embedding. The number of hidden units in bi-LSTM is 256. We train the model 100 epochs with a learning rate of 1×10^{-4} and a dropout probability of 0.35. The test accuracy is 61.2%. We break down the recall to each class label, as is listed in Table 4.1.

Table 4.1: The breakdown of recall on each class label in test data.

Label	**Total sentences**	**Correct classifications**	**Recall (%)**
Positive	909	637	70.1
Negative	912	587	64.4
Neutral	389	128	32.9

4.2 Sentiment Decomposition

In order to illustrate the ability of explainable methods, we visualize the relevance distributions via heat-maps over "positive" sentences (Figure 4.1) and "negative" sentences (Figure 4.2). In these heat-maps, we decompose the sentiment onto each word in a sentence, where positive relevance is mapped to red and negative relevance is mapped to blue. To this end, we conduct the decomposition experiment using the following explainable methods: *sensitivity analysis (SA)*, ϵ-LRP-all, ϵ-LRP-$relative$, ϵ-LPR-abs, β-LRP-all and abs-LRP-all.

From the observation of these heat-maps, we first notice that SA is not able to distinguish the words that have positive or negative contributions to the predicted label. SA distributes a relatively high relevance to words that express negative sentiment, such as "melodramatic" (positive example 1), "difficult" (positive example 2), "not", "odd" (positive example 3), while the output label is "positive"; or assigns a high relevance to positive words, such as "funny" and "provocative" (negative example 2), while the output label is "negative".

On the other hand, LRP is capable of recognizing the positive or negative words according to predictions. We notice that different LRP methods can generate different relevance distributions

for a sentence. From our investigation, ϵ-LRP-all produces the most reliable relevance distribution among these methods. It can successfully detect the words expressing positive sentiment, such as "effective", "thoughtful", "honesty", "beauty", "good", "funny" and "entertaining", and the words indicating negative sentiment, such as "difficult", "odd", "stupid", "maudlin", "dull", in different semantic environment. Another interesting property we observe is that ϵ-LRP-all method can recognize the sentiment negation. For example, in positive example 3, it discerns the opposite sentiment between "not" and "classic". In negative example 2, it highlights "neither" and "nor" as positive relevance while featuring "funny" and "provocative" as negative relevance towards the prediction.

Moreover, ϵ-LRP-$relative$ and ϵ-LPR-abs produce confusing relevance distributions that are not consistent with human intuition. β-LRP-all and abs-LRP-all perform better than ϵ-LRP-$relative$ and ϵ-LPR-abs, but a bit worse than the ϵ-LRP-all method. We notice that both β-LRP-all and abs-LRP-all can recognize the negation (positive example 3 and negative example 2), but fail to assign a negative relevance to "melodramatic" in positive example 1. Another issue is that β-LRP-all falsely considers "effective" as a negative word in sentence 1 of positive examples.

Method	ID	Label	Sentence
SA	1	positive	occasionally melodramatic , it 's also extremely effective .
	2	positive	a thoughtful , moving piece that faces difficult issues with honesty and beauty .
	3	positive	it 's a good film -- not a classic , but odd , entertaining and authentic .
ε–LRP–all	1	positive	occasionally melodramatic , it 's also extremely effective .
	2	positive	a thoughtful , moving piece that faces difficult issues with honesty and beauty .
	3	positive	it 's a good film -- not a classic , but odd , entertaining and authentic .
ε–LRP–relative	1	positive	occasionally melodramatic , it 's also extremely effective .
	2	positive	a thoughtful , moving piece that faces difficult issues with honesty and beauty .
	3	positive	it 's a good film -- not a classic , but odd , entertaining and authentic .
ε–LRP–abs	1	positive	occasionally melodramatic , it 's also extremely effective .
	2	positive	a thoughtful , moving piece that faces difficult issues with honesty and beauty .
	3	positive	it 's a good film -- not a classic , but odd , entertaining and authentic .
β–LRP–all	1	positive	occasionally melodramatic , it 's also extremely effective .
	2	positive	a thoughtful , moving piece that faces difficult issues with honesty and beauty .
	3	positive	it 's a good film -- not a classic , but odd , entertaining and authentic .
abs–LRP–all	1	positive	occasionally melodramatic , it 's also extremely effective .
	2	positive	a thoughtful , moving piece that faces difficult issues with honesty and beauty .
	3	positive	it 's a good film -- not a classic , but odd , entertaining and authentic .

Figure 4.1: Heat-maps of explainable methods for "positive" sentences.
(Red color represents positive relevance, and blue color represents negative relevance.
Color intensity is normalized based on absolute relevance values.)

Method	ID	Label	Sentence
SA	1	negative	after that , it just gets stupid and maudlin .
	2	negative	all this turns out to be neither funny nor provocative – only dull .
	3	negative	it 's just incredibly dull .
ε–LRP–all	1	negative	after that , it just gets stupid and maudlin .
	2	negative	all this turns out to be neither funny nor provocative – only dull .
	3	negative	it 's just incredibly dull .
ε–LRP–relative	1	negative	after that , it just gets stupid and maudlin .
	2	negative	all this turns out to be neither funny nor provocative – only dull .
	3	negative	it 's just incredibly dull .
ε–LRP–abs	1	negative	after that , it just gets stupid and maudlin .
	2	negative	all this turns out to be neither funny nor provocative – only dull .
	3	negative	it 's just incredibly dull .
β–LRP–all	1	negative	after that , it just gets stupid and maudlin .
	2	negative	all this turns out to be neither funny nor provocative – only dull .
	3	negative	it 's just incredibly dull .
abs–LRP–all	1	negative	after that , it just gets stupid and maudlin .
	2	negative	all this turns out to be neither funny nor provocative – only dull .
	3	negative	it 's just incredibly dull .

Figure 4.2: Heat-maps of explainable methods for "negative" sentences.
(Red color represents positive relevance, and blue color represents negative relevance.
Color intensity is normalized based on absolute relevance values.)

4.3 Representative Words

In order to evaluate the performance of explainable methods, we collect the representative words of each class label over the entire test dataset. Similar to the decomposition experiment, we apply the following six methods on all test sentences: *sensitivity analysis (SA)*, ϵ-LRP-all, ϵ-LRP-$relative$, ϵ-LPR-abs, β-LRP-all and abs-LRP-all.

4.3.1 "Positive" Label

For the test sentences classified as "positive" label, we use the explainable methods to compute relevance score of each word. Then we sort these words in the descending order of their relevance scores. Table 4.2 and Table 4.3 present the top ten most relevant words and least relevant words identified by these explainable methods for the positive label. From the word list of sensitivity analysis, we can see that the most relevant words are not directly expressing a positive attitude, while the least relevant words are mostly stop words, such as "the", "is", "that". On the other hand, from the LRP word list, the explanations behave differently if we use different strategies. Comparing to the zero-one distribution on multiplicative connections, the "relative" and "absolute" method cannot capture the words with strong positive sentiment effectively. One possible reason to this phenomenon is that the gate neuron is used to determine what fraction of information in source neuron should be passed to the next time step. If we back propagate the relevance to both gate neuron and source neuron, we actually pass redundant relevance message to the lower-layer neurons. Thus, the input neurons receive disturbing relevance and cannot capture the words that are correctly associated with the target label.

Moreover, from the least relevant word lists of ϵ-LRP-all, β-LRP-all and abs-LRP-all, we can observe that only the ϵ-LRP-all method can capture the words that are strongly against the "positive" sentiment, while the other two methods have disturbing symbols, such as question mark ("?") and ellipsis ("...").

Table 4.2: Ten most relevant words identified by the explainable methods over all test sentences for "Positive" label.

SA	ϵ-**LRP-**all	ϵ-**LRP-**$relative$	ϵ-**LRP-**abs	β-**LRP-**all	abs-**LRP-**all
authentic	authentic	and	well	deliciously	ingenious
feels	space	drama	for	ingenious	deliciously
sounds	invigorating	based	of	elegant	soulful
zippy	happy	-	a	wonderful	captivating
movie	enjoyable	well	love	refreshingly	wonderfully
counterculture	refreshing	on	and	pleasing	spectacular
60s	imaginative	of	based	refreshing	refreshingly
of	heartwarming	george	hilarious	captivating	pleasing
flourishes	fun	the	on	astonishing	resourceful
imaginative	rhythm	better	shot	spectacular	astonishing

Table 4.3: Ten least relevant words identified by the explainable methods over all test sentences for "Positive" label.

SA	ϵ-**LRP-**all	ϵ-**LRP-**$relative$	ϵ-**LRP-**abs	β-**LRP-**all	abs-**LRP-**all
that	undermines	of	ya	tadpole	ol
to	numbing	it	a	bravura	comedy
of	convenient	as	story	ol	terrible
,	furious	story	touching	lrb	schmaltzy
the	depressing	often	tender	lazy	suspenser
-	feels	is	films	hilarious	adultery
at	clueless	true	,	or	.
a	forgettable	,	shines	?	...
is	whiny	touching	often	presents	movie
with	lacking	tender	true	.	?

4.3.2 "Negative" Label

Similar to the "positive" label, we follow the same procedure to extract the representative words across all the test sentences, but now we only consider the "negative" predicted label. Table 4.4 and Table 4.5 present the top ten most relevant words and least relevant words identified by these explainable methods for the negative label. Based on the observation, we can reach similar conclusions: SA cannot capture the words effectively that most related or least related to the negative sentiment. Besides, the relative and absolute strategies perform worse than the zero-one strategy on the multiplicative connection. However, in the least relevant word list (Table 4.5), we notice that

β-LRP-all and abs-LRP-all are now able to capture the words that express "positive" sentiment, such as "best", "enjoy", "excellent", "brilliant", "hilarious" and "compelling".

Table 4.4: Ten most relevant words identified by the explainable methods over all test sentences for "Negative" label.

SA	ϵ-**LRP-**all	ϵ-**LRP-**$relative$	ϵ-**LRP-**abs	β-**LRP-**all	abs-**LRP-**all
mothman	trite	film	the	stupid	lifeless
prophecies	futile	cute	that	lifeless	stupid
trite	lifeless	ideas	idea	flimsy	flimsy
unk	overproduced	too	incoherent	waste	incoherent
psychological	slip	stupid	a	shoddy	shoddy
difficult	formulaic	be	dreary	bomb	waste
ramble	incoherent	surrounding	potter	pile	unfocused
makers	patchwork	few	arts	lousy	bomb
divine	inconsequential	to	imagine	incoherent	mess
someone	soggy	although	for	mess	lousy

Table 4.5: Ten least relevant words identified by the explainable methods over all test sentences for "Negative" label.

SA	ϵ-**LRP-**all	ϵ-**LRP-**$relative$	ϵ-**LRP-**abs	β-**LRP-**all	abs-**LRP-**all
with	sisters	trying	moviegoers	roles	.
unk	imax	mystery	martial	warmed	excellent
a	guessing	incoherent	the	enjoy	brilliant
...	definitely	chuckles	of	are	capture
which	refreshing	,	ideas	peace	hilarious
hours	blend	satire	-	reconciliation	roles
on	delicate	several	film	lrb	documentary
of	captures	and	cute	capture	peace
for	vibrant	has	some	best	compelling
two	enjoyed	a	fate	documentary	best

4.3.3 "Neutral" Label

Table 4.6 and Table 4.7 list the ten most relevant and the ten least relevant words for the "neutral" label, respectively. We notice that none of the explainable methods can capture the neutral words properly. We speculate that the possible reasons for this phenomenon are three-fold: neutrality is difficult to define; there are limited number of words in the test dataset; both positive words and negative words may appear in one sentence to express neutrality. Thus, we can either use a larger dataset or build a more robust model in order to capture the neutral words.

Table 4.6: Ten most relevant words identified by the explainable methods over all test sentences for "Neutral" label.

SA	ϵ-**LRP-**all	ϵ-**LRP-**$relative$	ϵ-**LRP-**abs	β-**LRP-**all	abs-**LRP-**all
version	birot	sensual	it	or	or
mgm	preciousness	abbass	sensual	gangs	undercover
rethink	intentions	performance	the	what	bet
object	numbers	the	's	bet	gangs
films	damn	it	difficult	undercover	birot
excellent	guard	drama	drama	passably	passably
madcap	terrible	story	of	largely	.
nazi	mediocre	her	contemporary	ozpetek	views
women	constantly	rapidly	beyond	deviant	largely
cult	solid	good	get	ha	ha

Table 4.7: Ten least relevant words identified by the explainable methods over all test sentences for "Neutral" label.

SA	ϵ-**LRP-**all	ϵ-**LRP-**$relative$	ϵ-**LRP-**abs	β-**LRP-**all	abs-**LRP-**all
,	young	artists	flimsy.	fairly	really
-	explosion	whose	the	what	it
female	rethink	too	,	remember	fairly
to	fresh	teasing	abbass	too	there
it	still	its	to	you	i
third	excellent	but	from	claude	no
is	paved	lrb	a	no	leave
four	genre	flimsy	performance	we	we
about	artistically	rrb	lrb	yes	like
the	twinkle	from	rrb	i	what

4.4 Word Perturbation Experiemnt

In this section, we perform a word perturbation experiment in order to validate the word-level relevance of explainable methods. In these experiments, we only consider sentences with length greater or equal to 10 words in the test dataset. This results in 1,880 sentences in total. For each of these sentences, we delete up to 5 words according to their relevance scores. Here when we delete a word in a sentence, we set its word embedding to a zero vector in the input sentence representation. After deleting words, we re-input the sentences to the bi-LSTM model, and obtain their sentiment classifications. The intuition behind this experiment is that an explainable method should reveal the words that are important to the model's decision. Therefore, if we delete the words based on their relevance scores, the model may predict different labels. For example, suppose we have a sentence "it never fails to be a great movie" and the model classifies it as "positive". If we remove the word "fails", the model may change its decision and predict "negative".

We perform the word deletion experiment on two categories of sentences. For those sentences that are initially correctly classified, we delete words in the descending order of their relevances. The idea behind this approach is that words with higher relevances should have more positive contributions on the predictions. If we delete these words sequentially, the model may change its prediction at some point. For those sentences that are falsely classified, we delete words in the descending order of their relevances as well. The intuition is that the model may focus on inappropriate words when it classifies a sentence incorrectly. That is, the model assigns a high relevance score to the word that is not supposed to be important to its decision.

Moreover, we conduct a random deletion experiment as comparison. That is, we randomly delete words for each test sentence, and re-predict via the bi-LSTM model. For each number of deletion, we average the results over 8 runs.

Figure 4.3 presents the classification accuracy with respect to the number of deleted words. As we

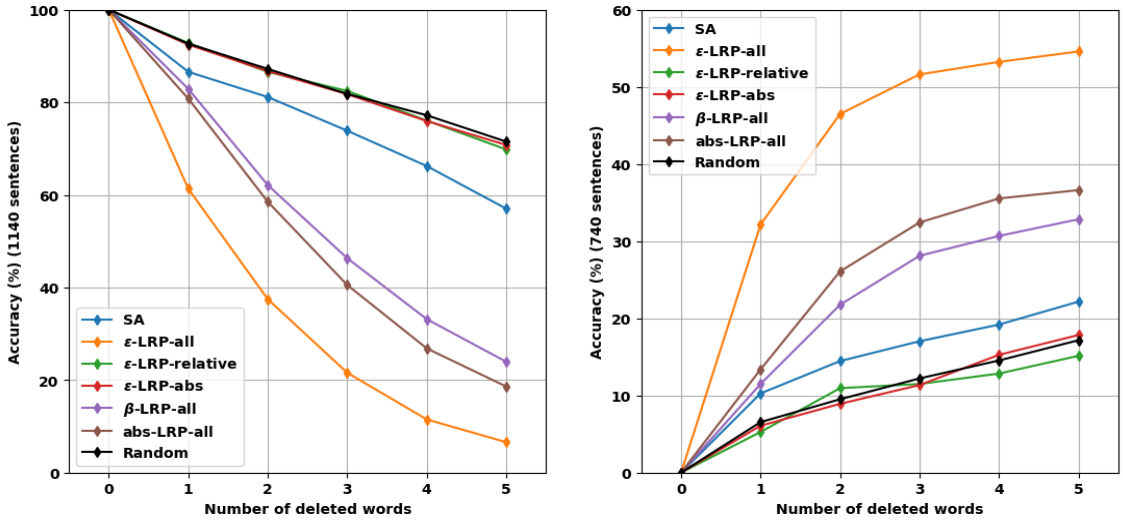

Figure 4.3: Effect of word deletion on initially correctly (left) and falsely (right) classified sentences using different LRP rules.

expected, the classification accuracy decreases as we delete more relevant words on the initially correct sentences (Figure 4.3 left), while the classification accuracy increases as we delete more disturbing words (Figure 4.3 right). According to our intuition, a steeper curve of accuracy represents a better relevance distribution. We observe that the ϵ-LRP-$relative$ and ϵ-LRP-abs methods perform closely with the random strategy. This indicate these two explainable methods fail to identify the words that should be important to the model's decision. It also justifies their poor performance when capturing the representative words for different classes. Moreover, we notice that the ϵ-LRP-all, β-LRP-all and abs-LRP-all methods perform better than the gradient-based sensitivity analysis. Among these three methods, ϵ-LRP-all performs the best in this sentiment classification task. However, we need to point out that explainable methods may have different behaviors when applied on different model architectures. Bharadhwaj [39] reported that β-LRP-all outperforms ϵ-LRP-all on RNN, while Arras et al. [102] showed that the ϵ-rule performs better on CNN.

4.5 Case study

In this section, we focus on two classification cases and use explainable method to interpret the model behavior under different circumstance. Here we use the ϵ-LRP-all method, since it has the best performance in this task.

Figure 4.4 illustrates the heat-maps of two sentences, where color with high intensity represents a high absolute relevance score. For sentence 1, the model successfully capture the word "funny" as the most relevant word for positive sentiment. Then we add three negation words or phrases to the left of the sentence, i.e., "fail to be", "not", "never". Surprisingly, only "fail to be" and "not" turn the sentiment into negative (1-a, 1-b), while the model does not consider "never" as a negation (1-c). The underlying reason is that the model does not treat "never" as a strong negative word, such that the strong positive word "funny" overshadows the word "never". Next a double negation is added in front the sentence (1-d). We can see that the model fails to transform the double negation into the positive sentiment. We speculate that the model regards "fail" as a much stronger negative word than "not", so that it makes the decision only based on "fail" and "funny".

For sentence 2, the model labels it as positive mainly based on "daring", "mesmerizing" and "hard to forget". However, if we only input "exceedingly hard to forget" (2-a), the model falsely classified it as negative, because the model is inefficient in handling double negation. Then we change "hard" and "forget" to "easy" and "remember" respectively (2-a ~2-d), and re-predict via the model. We find out that the model can successfully treat "forget" as negative word and "remember" as positive word in different scenarios. The problem lies in the word "hard". We notice that the model cannot handle "hard" correctly when it is followed by "forget" and "remember" (2-a, 2-b). In contrast, it can distinguish the meaning of "easy" under different semantic environment (2-c, 2-d).

Both cases illustrate the complex mechanism involved in semantic decomposition. Due to the limited data, the model may have a bias towards the training sentences. In other words, the model

makes its decision completely depending on its own knowledge.

ID	Prediction	Sentence
1	positive	a very funny movie .
1–a	negative	fail to be a very funny movie .
1–b	negative	not a very funny movie .
1–c	positive	never a very funny movie .
1–d	negative	not fail to be a very funny movie .
2	positive	daring , mesmerizing and exceedingly hard to forget .
2–a	negative	exceedingly hard to forget
2–b	positive	exceedingly hard to remember
2–c	negative	exceedingly easy to forget
2–d	positive	exceedingly easy to remember

Figure 4.4: Test sentences (1, 2) and manually modified sentences (1-a ~1-d, 2-a ~2-d). (red: positive relevance, blue: negative relevance)

5. CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

In this work, we propose novel techniques to explain the recurrent neural network with LSTM. Considering the weighted linear and multiplicative connections in LSTM, we extend the existing Layer-wise Relevance Propagation (LRP) framework with different strategies on the two types of connections. For the weighted linear connections, we use ϵ-rule, β-rule and abs-rule. Specifically, we propose a new approach to determine the β value based on each set of linear connections in β-rule, and a new abs-rule that passes the relevance only depending on the absolute weighted connections. For the multiplicative connections, we distribute the relevance of output neuron back to gate neuron and source neuron using $relative$ distribution and $absolute$ distribution. Moreover, we implement the "zero-one" distribution on the multiplicative connections.

Next, we build a Bi-LSTM model for the sentiment classification task. The model is trained on the SST movie review dataset, and the test accuracy reaches 61.2%. Then we implement these explainable methods on the Bi-LSTM model. The methods we tested are: gradient-based *sensitivity analysis (SA)*, ϵ-LRP-all, ϵ-LRP-$relative$, ϵ-LPR-abs, β-LRP-all and abs-LRP-all.

There are four word-level experiments we perform to evaluate our explainable methods: sentiment decomposition, top representative words collection, word perturbation and case study. The results reveal that the ϵ-LRP-all method outperforms the other methods in the sentiment analysis task. More specifically, comparing to the other methods, ϵ-LRP-all is able to decompose a reasonable relevance distribution onto each input word, detect negation patterns in text data, collect reliable representative words for each class label, and explain the Bi-LSTM model in a human understandable way.

Moreover, we need to point out that these explainable methods may have different behaviors in different neural network architectures. Therefore, it is a good practice for researchers to adapt the explainable methods to their specific applications.

5.2 Future Work

Our future work includes the following three directions:

- Extend the β-rule and propose strategies to assign the β value to each set of linear connections.
- Evaluate the explainable methods on different NLP applications, such as question-answering and speech recognition.
- Optimize the architecture of LSTM to reach better model performance.

REFERENCES

[1] A. Ng, "What data scientists should know about deep learning," *URL https://www. slideshare. net/ExtractConf*, vol. 44, 2015.

[2] A. Fattah, "The limits of machine learning – is your ml solution viable?," *URL https://www.linkedin.com/pulse/limits-machine-learning-your-ml-solution-viable-ahmed-fattah/*, 2017.

[3] M. A. Nielsen, *Neural networks and deep learning*, vol. 2018. Determination press San Francisco, CA, 2015.

[4] S. Yan, "Understanding lstm networks," *Online). Accessed on August*, vol. 11, 2015.

[5] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 6645–6649, IEEE, 2013.

[6] R. C. Deo, "Machine learning in medicine," *Circulation*, vol. 132, no. 20, pp. 1920–1930, 2015.

[7] R. Culkin and S. R. Das, "Machine learning in finance: The case of deep learning for option pricing," *Journal of Investment Management*, vol. 15, no. 4, pp. 92–100, 2017.

[8] S. Schelter, F. Biessmann, T. Januschowski, D. Salinas, S. Seufert, and G. Szarvas, "On challenges in machine learning model management," 2018.

[9] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020.

[10] P.-C. Yang, K. Sasaki, K. Suzuki, K. Kase, S. Sugano, and T. Ogata, "Repeatable folding task by humanoid robot worker using deep learning," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 397–403, 2016.

[11] L. Deng, G. Hinton, and B. Kingsbury, "New types of deep neural network learning for speech recognition and related applications: An overview," in *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 8599–8603, IEEE, 2013.

[12] A. Korotcov, V. Tkachenko, D. P. Russo, and S. Ekins, "Comparison of deep learning with multiple machine learning methods and metrics using diverse drug discovery data sets," *Molecular pharmaceutics*, vol. 14, no. 12, pp. 4462–4475, 2017.

[13] S. Liu, S. Liu, W. Cai, S. Pujol, R. Kikinis, and D. Feng, "Early diagnosis of alzheimer's disease with deep learning," in *2014 IEEE 11th international symposium on biomedical imaging (ISBI)*, pp. 1015–1018, IEEE, 2014.

[14] D. Gunning, "Explainable artificial intelligence (xai)," *Defense Advanced Research Projects Agency (DARPA), nd Web*, vol. 2, p. 2, 2017.

[15] H. Liu, Q. Yin, and W. Y. Wang, "Towards explainable nlp: A generative explanation framework for text classification," *arXiv preprint arXiv:1811.00196*, 2018.

[16] E. Oduor, K. Qian, Y. Li, and L. Popa, "Xait: An interactive website for explainable ai for text," in *Proceedings of the 25th International Conference on Intelligent User Interfaces Companion*, pp. 120–121, 2020.

[17] W. Samek, T. Wiegand, and K.-R. Müller, "Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models," *arXiv preprint arXiv:1708.08296*, 2017.

[18] J. R. Zilke, E. L. Mencía, and F. Janssen, "Deepred–rule extraction from deep neural networks," in *International Conference on Discovery Science*, pp. 457–473, Springer, 2016.

[19] M. Sato and H. Tsukimoto, "Rule extraction from neural networks via decision tree induction," in *IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222)*, vol. 3, pp. 1870–1875, IEEE, 2001.

[20] Z. Che, S. Purushotham, R. Khemani, and Y. Liu, "Interpretable deep models for icu outcome prediction," in *AMIA Annual Symposium Proceedings*, vol. 2016, p. 371, American Medical Informatics Association, 2016.

[21] M. D. Zeiler, G. W. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," in *2011 International Conference on Computer Vision*, pp. 2018–2025, IEEE, 2011.

[22] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*, pp. 818–833, Springer, 2014.

[23] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921–2929, 2016.

[24] L. Arras, G. Montavon, K.-R. Müller, and W. Samek, "Explaining recurrent neural network predictions in sentiment analysis," *arXiv preprint arXiv:1706.07206*, 2017.

[25] A. Karpathy, J. Johnson, and L. Fei-Fei, "Visualizing and understanding recurrent networks," *arXiv preprint arXiv:1506.02078*, 2015.

[26] S. Wisdom, T. Powers, J. Pitton, and L. Atlas, "Interpretable recurrent neural networks using sequential sparse recovery," *arXiv preprint arXiv:1611.07252*, 2016.

[27] Q. Zhang, Y. Nian Wu, and S.-C. Zhu, "Interpretable convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8827–8836, 2018.

[28] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PloS one*, vol. 10, no. 7, p. e0130140, 2015.

[29] R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, *et al.*, "Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 618–626.

[30] Y. Ding, Y. Liu, H. Luan, and M. Sun, "Visualizing and understanding neural machine translation," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1150–1159, 2017.

[31] L. Arras, J. Arjona-Medina, M. Widrich, G. Montavon, M. Gillhofer, K.-R. Müller, S. Hochreiter, and W. Samek, "Explaining and interpreting lstms," in *Explainable ai: Interpreting, explaining and visualizing deep learning*, pp. 211–238, Springer, 2019.

[32] M. Faruqui and C. Dyer, "Improving vector space word representations using multilingual correlation," in *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 462–471, 2014.

[33] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[34] V. Yadav and S. Bethard, "A survey on recent advances in named entity recognition from deep learning models," *arXiv preprint arXiv:1910.11470*, 2019.

[35] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[36] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1253, 2018.

[37] J. Li, W. Monroe, and D. Jurafsky, "Understanding neural networks through representation erasure," *arXiv preprint arXiv:1612.08220*, 2016.

[38] L. Arras, A. Osman, K.-R. Müller, and W. Samek, "Evaluating recurrent neural network explanations," *arXiv preprint arXiv:1904.11829*, 2019.

[39] H. Bharadhwaj, "Layer-wise relevance propagation for explainable deep learning based speech recognition," in *2018 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pp. 168–174, IEEE, 2018.

[40] J. Dean, "Large-scale deep learning for building intelligent computer systems," 2016.

[41] D. Wang and J. Chen, "Supervised speech separation based on deep learning: An overview," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 10, pp. 1702–1726, 2018.

[42] K. Makantasis, K. Karantzalos, A. Doulamis, and N. Doulamis, "Deep supervised learning for hyperspectral data classification through convolutional neural networks," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 4959–4962, IEEE, 2015.

[43] M. Längkvist, L. Karlsson, and A. Loutfi, "A review of unsupervised feature learning and deep learning for time-series modeling," *Pattern Recognition Letters*, vol. 42, pp. 11–24, 2014.

[44] Y. Bengio, "Deep learning of representations for unsupervised and transfer learning," in *Proceedings of ICML workshop on unsupervised and transfer learning*, pp. 17–36, 2012.

[45] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[46] E. Yazan and M. F. Talu, "Comparison of the stochastic gradient descent based optimization techniques," in *2017 International Artificial Intelligence and Data Processing Symposium (IDAP)*, pp. 1–5, IEEE, 2017.

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

[48] P. Wang, Y. Qian, F. K. Soong, L. He, and H. Zhao, "A unified tagging solution: Bidirectional lstm recurrent neural network with word embedding," *arXiv preprint arXiv:1511.00215*, 2015.

[49] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[50] M. Rusinol and J. Llados, "Logo spotting by a bag-of-words approach for document categorization," in *2009 10th international conference on document analysis and recognition*, pp. 111–115, IEEE, 2009.

[51] S. Ma, X. Sun, Y. Wang, and J. Lin, "Bag-of-words as target for neural machine translation," *arXiv preprint arXiv:1805.04871*, 2018.

[52] L. Purda and D. Skillicorn, "Accounting variables, deception, and a bag of words: Assessing the tools of fraud detection," *Contemporary Accounting Research*, vol. 32, no. 3, pp. 1193–1223, 2015.

[53] J. Ramos *et al.*, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the first instructional conference on machine learning*, vol. 242, pp. 133–142, New Jersey, USA, 2003.

[54] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[55] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, pp. 3111–3119, 2013.

[56] J. Lilleberg, Y. Zhu, and Y. Zhang, "Support vector machines and word2vec for text classification with semantic features," in *2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC)*, pp. 136–140, IEEE, 2015.

[57] S. K. Sienčnik, "Adapting word2vec to named entity recognition," in *Proceedings of the 20th Nordic Conference of Computational Linguistics (NODALIDA 2015)*, pp. 239–243, 2015.

[58] B. Xue, C. Fu, and Z. Shaobin, "A study on sentiment computing and classification of sina weibo with word2vec," in *2014 IEEE International Congress on Big Data*, pp. 358–363, IEEE, 2014.

[59] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[60] F. Dernoncourt, J. Y. Lee, and P. Szolovits, "Neuroner: an easy-to-use program for named-entity recognition based on neural networks," *arXiv preprint arXiv:1705.05487*, 2017.

[61] J. Abdillah, I. Asror, Y. F. A. Wibowo, *et al.*, "Emotion classification of song lyrics using bidirectional lstm method with glove word representation weighting," *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, vol. 4, no. 4, pp. 723–729, 2020.

[62] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, "Deep learning for hate speech detection in tweets," in *Proceedings of the 26th International Conference on World Wide Web Companion*, pp. 759–760, 2017.

[63] B. Liu, "Sentiment analysis and opinion mining," *Synthesis lectures on human language technologies*, vol. 5, no. 1, pp. 1–167, 2012.

[64] D. Yang, D. Zhang, Z. Yu, and Z. Wang, "A sentiment-enhanced personalized location recommendation system," in *Proceedings of the 24th ACM conference on hypertext and social media*, pp. 119–128, 2013.

[65] P. V. Krishna, S. Misra, D. Joshi, and M. S. Obaidat, "Learning automata based sentiment analysis for recommender system on cloud," in *2013 International Conference on Computer, Information and Telecommunication Systems (CITS)*, pp. 1–5, IEEE, 2013.

[66] Y. Wang, M. Wang, and W. Xu, "A sentiment-enhanced hybrid recommender system for movie recommendation: a big data analytics framework," *Wireless Communications and Mobile Computing*, vol. 2018, 2018.

[67] T. Silva and M. Fernando, "Knowledge based approach for concept level sentiment analysis for online reviews,"

[68] D. Rajagopal, E. Cambria, D. Olsher, and K. Kwok, "A graph-based approach to commonsense concept extraction and semantic similarity detection," in *Proceedings of the 22nd International Conference on World Wide Web*, pp. 565–570, 2013.

[69] H. Kang, S. J. Yoo, and D. Han, "Senti-lexicon and improved naïve bayes algorithms for sentiment analysis of restaurant reviews," *Expert Systems with Applications*, vol. 39, no. 5, pp. 6000–6010, 2012.

[70] A. Prabhat and V. Khullar, "Sentiment classification on big data using naïve bayes and logistic regression," in *2017 International Conference on Computer Communication and Informatics (ICCCI)*, pp. 1–5, IEEE, 2017.

[71] M. Ahmad, S. Aftab, and I. Ali, "Sentiment analysis of tweets using svm," *Int. J. Comput. Appl*, vol. 177, no. 5, pp. 25–29, 2017.

[72] M. Bilal, H. Israr, M. Shahid, and A. Khan, "Sentiment classification of roman-urdu opinions using naïve bayesian, decision tree and knn classification techniques," *Journal of King Saud University-Computer and Information Sciences*, vol. 28, no. 3, pp. 330–344, 2016.

[73] S. Liao, J. Wang, R. Yu, K. Sato, and Z. Cheng, "Cnn for situations understanding based on sentiment analysis of twitter data," *Procedia computer science*, vol. 111, pp. 376–381, 2017.

[74] D. Li and J. Qian, "Text sentiment analysis based on long short-term memory," in *2016 First IEEE International Conference on Computer Communication and the Internet (ICCCI)*, pp. 471–475, IEEE, 2016.

[75] X. Wang, W. Jiang, and Z. Luo, "Combination of convolutional and recurrent neural network for sentiment analysis of short texts," in *Proceedings of COLING 2016, the 26th international conference on computational linguistics: Technical papers*, pp. 2428–2437, 2016.

[76] A. Mukwazvure and K. Supreethi, "A hybrid approach to sentiment analysis of news comments," in *2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO)(Trends and Future Directions)*, pp. 1–6, IEEE, 2015.

[77] O. Appel, F. Chiclana, J. Carter, and H. Fujita, "A hybrid approach to the sentiment analysis problem at the sentence level," *Knowledge-Based Systems*, vol. 108, pp. 110–124, 2016.

[78] G. Yoo and J. Nam, "A hybrid approach to sentiment analysis enhanced by sentiment lexicons and polarity shifting devices," in *The 13th Workshop on Asian Language Resources*, pp. 21–28, 2018.

[79] C. Grosan and A. Abraham, *Intelligent systems*. Springer, 2011.

[80] D. Doran, S. Schulz, and T. R. Besold, "What does explainable ai really mean? a new conceptualization of perspectives," 2017.

[81] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, *et al.*, "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information Fusion*, vol. 58, pp. 82–115, 2020.

[82] Z. Bursac, C. H. Gauss, D. K. Williams, and D. W. Hosmer, "Purposeful selection of variables in logistic regression," *Source code for biology and medicine*, vol. 3, no. 1, p. 17, 2008.

[83] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*, vol. 398. John Wiley & Sons, 2013.

[84] C.-Y. J. Peng, K. L. Lee, and G. M. Ingersoll, "An introduction to logistic regression analysis and reporting," *The journal of educational research*, vol. 96, no. 1, pp. 3–14, 2002.

[85] O. Sagi and L. Rokach, "Explainable decision forest: Transforming a decision forest into an interpretable tree," *Information Fusion*, 2020.

[86] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, "An knn model-based approach and its application in text categorization," in *International Conference on Intelligent Text Processing and Computational Linguistics*, pp. 559–570, Springer, 2004.

[87] S. Jiang, G. Pang, M. Wu, and L. Kuang, "An improved k-nearest-neighbor algorithm for text categorization," *Expert Systems with Applications*, vol. 39, no. 1, pp. 1503–1509, 2012.

[88] S.-K. Min, D. Simonis, and A. Hense, "Probabilistic climate change predictions applying bayesian model averaging," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 365, no. 1857, pp. 2103–2116, 2007.

[89] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.

[90] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.

[91] R. G. Cinbis, J. Verbeek, and C. Schmid, "Weakly supervised object localization with multi-fold multiple instance learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 1, pp. 189–203, 2016.

[92] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1717–1724, 2014.

[93] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Is object localization for free?-weakly-supervised learning with convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 685–694, 2015.

[94] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

[95] H. Strobelt, S. Gehrmann, H. Pfister, and A. M. Rush, "Lstmvis: A tool for visual analysis of hidden state dynamics in recurrent neural networks," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 1, pp. 667–676, 2017.

[96] J. Li, X. Chen, E. Hovy, and D. Jurafsky, "Visualizing and understanding neural models in nlp," *arXiv preprint arXiv:1506.01066*, 2015.

[97] W. J. Murdoch, P. J. Liu, and B. Yu, "Beyond word importance: Contextual decomposition to extract interactions from lstms," *arXiv preprint arXiv:1801.05453*, 2018.

[98] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, "Network dissection: Quantifying interpretability of deep visual representations," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6541–6549, 2017.

[99] M. T. Ribeiro, S. Singh, and C. Guestrin, "" 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*, pp. 1135–1144, 2016.

[100] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital Signal Processing*, vol. 73, pp. 1–15, 2018.

[101] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller, "Evaluating the visualization of what a deep neural network has learned," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 11, pp. 2660–2673, 2016.

[102] L. Arras, F. Horn, G. Montavon, K.-R. Müller, and W. Samek, "Explaining predictions of non-linear classifiers in nlp," *arXiv preprint arXiv:1606.07298*, 2016.