Natural Language Processing: Sentiment Analysis

FangRui

November 2021

Abstract

This work is a simple implementation of a deep learning model consisting of three components: *Word Embedding*, *Encoder* and *Classifier*. The model is based on a **pytorch** implementation. Different word embedding methods: *Random Initialisation, Word2Vec, GloVe* and different encoders: *RNN, LSTM, GRU* – are tried in the model during the experiments.

The report is divided into five main sections, followed by a description of the main processing modules used in the experiments(**Section I**), the dataset(**Section II**), the extra module and training strategies(**Section III**), modules comparison (**Section IV**), and the tips gained from the experiments(**Section V**).

1 Introduction

1.1 Word embedding

1.1.1 Random Initialization

Random Initialization randomly initializes a simple lookup table that stores embeddings of a fixed dictionary and size.

1.1.2 Word2Vector

Word2vec algorithm uses a neural network model to learn word associations from a large corpus of text. Once trained, such a model can detect synonymous words or suggest additional words for a partial sentence. As the name implies, word2vec represents each distinct word with a particular list of numbers called a vector. The vectors are chosen carefully such that a simple mathematical function (the cosine similarity between the vectors) indicates the level of semantic similarity between the words represented by those vectors.

1.1.3 GloVe

GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

1.2 Encoder

1.2.1 RNN

Recurrent neural network (RNN) is a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence. This allows it to exhibit temporal dynamic behavior. Derived from feedforward neural networks, RNNs can use their internal state (memory) to process variable length sequences of inputs.

1.2.2 LSTM

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell.

LSTMs were developed to deal with the vanishing gradient problem that can be encountered when training traditional RNNs. Relative insensitivity to gap length is an advantage of LSTM over RNNs, hidden Markov models and other sequence learning methods in numerous applications.

1.2.3 GRU

GRU is the newer generation of Recurrent Neural networks and is pretty similar to an LSTM. GRU's got rid of the cell state and used the hidden state to transfer information. It also only has two gates, a reset gate and update gate.

2 Dataset

The Stanford Sentiment Treebank is a corpus with fully labeled parse trees that allows for a complete analysis of the compositional effects of sentiment in language. The corpus is based on the dataset introduced by Pang and Lee (2005) and consists of 11,855 single sentences extracted from movie reviews. It was parsed with the Stanford parser and includes a total of 215,154 unique phrases from those parse trees, each annotated by 3 human judges.

Each phrase is labelled as either negative, somewhat negative, neutral, somewhat positive or positive. The corpus with all 5 labels is referred to as SST-5 or SST fine-grained. Binary classification experiments on full sentences (negative or somewhat negative vs somewhat positive or positive with neutral sentences discarded) refer to the dataset as SST-2 or SST binary.

3 Extra module and Training Strategy

3.1 Attention

Attention is a technique that mimics cognitive attention. The effect enhances the important parts of the input data and fades out the rest—the thought being that the network should devote more computing power to that small but important part of the data. Which part of the data is more important than others depends on the context and is learned through training data by gradient descent.

3.2 Training strategy

Learning rate decay and gradient clipping are used for better training and prevent overfitting

4 Comparison

Table 1 demonstrate the accuracy of different models on validation dataset of SST-2.

5 Tips

We use batch size = 16, learning rate = 10^{-4}, and weight decaying = 10^{-5}. For learning rate decaying, we made learning rate update as: $LR' =$

Models	SST-2 Accuracy
Random-BiLSTM	70.06
GloVe-BiLSTM	78.13
Word2Vec-BiLSTM	77.52
GloVe-GRU-Attn	83.01
GloVe-LSTM-Attn	**84.16**

Table 1: Validation accuracy on different models

$0.5^{\frac{epoch-threshold}{50}} * LR$, where threshold is the epoch starting learning rate decay.