Amazon Fine Food Reviews Sentiment Analysis with Recurrent Neural Network

Domain Background

With the emergence of both social media and online reviews, sentiment analysis became an important area of research in Machine Learning. Sentiment analysis can help improve products, marketing, clinical medicine or even how cities deliver services. Sentiment analysis is a very interesting challenge to work on because the computer needs to interact with human language (natural language processing) and be able to identify and extract subjective information.

Several algorithms like Naive Bayes, Support Vector Machine and Maximum Entropy are commonly used for sentiment analysis. However with the recent development of recurrent neural networks, Deep Learning has started to outperform all the other methods due to its ability to build a representation of whole sentences based on the sentence structure. Several amazing research papers have already been published on this topic by Stanford (e.g. Sentiment Analysis) and OpenAi (e.g. unsupervised sentiment neuron).

Problem Statement

For this project, we will predict if the Amazon Fine Food Reviews are positive or negative using a recurrent neural network. This is a supervised learning problem where we need to predict the positive or negative target variable for each review. The goal will be to maximize the accuracy of this classification. We will train our model on a dataset containing thousands of reviews presented as unstructured text. Each review will be labeled as positive or negative.

Datasets and Inputs

For this project, we will use the Amazon Fine Food Review from Kaggle. The dataset contains 568,454 food reviews Amazon users left up to October 2012.

The review for a specific product by a specific user is stored in the Text column. The Text column consists of an unstructured text. In addition to the Text feature representing the review, we can identify the product using the ProductId feature, the Amazon user with the UserID feature, the date the review was posted by the Time column, and a brief summary of the review with the Summary variable. Additionally, HelpfulnessNumerator and HelpfulnessDenominator, two variables representing how helpful a review is, are also provided in the review.

More importantly, for each review, we have a “Score” variable representing a rating from 1 to 5 (1 is a poor review, and 5 is an excellent review). To simplify the problem, we will consider that a review with a score of 4 or 5 is positive and a review with a score of 1 - 3 is negative.

Many very valuable features are provided in this dataset. However, to build our sentiment analysis model we will use the Text feature as an input and the Score feature as an output.

Solution Statement

In order to predict if a review is positive or negative we will use a recurrent neural network. Most of the methods for sentiment analysis look at each word individually, attributing positive points for positive words and negative points for negative words, and then total the points. This is the lexicon approach. However the problem with this method is that it ignores the order of the words, which can lead to the loss of important information. The Deep Learning approach can understand subtleties because it doesn’t analyze text at face value—it creates abstract representations of what it learned.

We will use Long Short Term Memory Networks (LSTM), which is a special case of RNN. The main advantage of LSTM is that it can memorize information. When the gap between two pieces information is too important, RNN becomes unable to learn to connect information. To learn more about LSTM, you can read this excellent article written by Christopher Olah. We will train this algorithm on the Amazon Fine Food Review.

Benchmark Model

To benchmark our model, we could compare the result of our model to the solutions provided by other Kagglers on the Kaggle forums.

For example, Guillaume Payen was able to achieve an F1 score of 0.92 using a traditional approach like Naive Bayes and logistic regression model. You can find his notebook here.

Evaluation Metrics

In order to evaluate the model, we will use the F1 score to test accuracy. It considers both precision and recall with the same weight. In the case of fine food reviews, it makes sense to put the same weight on precision and recall. However, in the case of clinical medicine, we might have put more weight on precision. The best F1 score is 1 and the worst is 0.

F1 = 2 * precision*recall / (precision + recall)

Project Design

To build our RRN we will use Tensorflow, which is one of the most popular frameworks for deep learning.

Data Preprocessing

The first step is to remove punctuation from the review, as it doesn't provide any information to determine if a review is positive or negative.

Encoding the words

Then we will have to transform words into integers. We can create a dictionary to map the vocabulary contained in the reviews to integers.

Encoding the labels

The label is a score between 1 and 5. We want to predict if a review is positive or not. We will consider a review positive if it is greater than or equal to 4, and negative if it is between 1 and 3. The neural network expect integers for labels, so we will convert a score between 1 and 3 to 0, and the score between 4 and 5 to 1.

Split the data into training, validation and testing

In order to build the RNN model, it's important to split the data into training, validation and testing datasets. We will train the model on a training set, and tune the model by validating the results on the validation dataset. Finally we will measure the ability of our model to generalize by testing our model on the testing dataset.

Embedding

Instead of using one-hot encoding, one efficient way to represent words is to use embeddings. Embedding consist of mapping words to small vectors. To generate these embeddings we will use the word2vec approach so when words have similar meanings they will be close to each other.

Build the LSTM Network

We will build the LSTM model using Tensorflow. The Tensorflow documentation for RRN can be found here. We will have to experiment with several network architectures to optimize accuracy, such as number of layers, whether or not there's a dropout layer, etc.

Train the model

The next step will be to train our model. We will use the stochastic gradient descent approach.This method consists of calculating an estimate of the loss function (Cross Entropy in our case) on small batches of the dataset. This approach is more scalable than traditional gradient descent.

In order to optimize the accuracy of our model we will have to tune the hyperparameters. The number of hyperparameters will depend on the architecture of the network but we can anticipate that we will have to tune at least the following parameters:

1. Number of neural network layers
2. Size of the neural network layers
3. Learning rate
4. Number of epochs
5. Batch size
6. Probability for the dropout layer
7. Etc.

Test the model

Finally, we will have to calculate the accuracy of our model to measure the ability of our model to generalize.

Optional next steps: generate fine food reviews with Recurrent Neural Network

If our neural network is able to classify if a review is positive or negative, that in and of itself is already mind blowing. But what if we were able to generate fine food reviews with a recurrent neural network? This would demonstrate that RRN can not only interpret sentiment but also write text like humans, and describe a product like wine or cheese. We can find an example of text generation on Andrej Karphaty blog.