TASK

Capstone Project III

Visit our website

Introduction

Welcome to the third Capstone Project Task!

For this Capstone project, you are required to use recurrent neural networks in Keras to try and classify a movie review as either positive or negative..

Get in touch

Connect for support

Remember that with our courses, you're not alone! You can contact your mentor to get support on any aspect of your course.

The best way to get help is to login to **www.hyperiondev.com/portal** to start a chat with your mentor. You can also schedule a call or get support via email.

Your mentor is happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!

Introduction to the Task

This Capstone Project is about a basic form of Natural Language Processing (NLP) called Sentiment Analysis. For this task you are required to use a recurrent neural network in Keras to try and classify a movie review as either positive or negative.

For example, consider the review, "This movie is not very good." This text ends with the words "very good" which indicates a very positive sentiment, but it is negated because it is preceded by the word "not", so the text should be classified as having a negative sentiment. We need to teach our neural network to recognise this distinction and be able to classify the review correctly.

This problem can be broken down into the following steps:

1. Get the dataset
2. Preprocessing the Data
3. Build the Model
4. Train the model
5. Test the Model
6. Predict Something

Get the Dataset

For this task we will be using the Large Movie View Dataset. This dataset contains a collection of 50,000 reviews from IMDb and contains an even number of positive and negative reviews. A negative review has a score ≤ 4 out of 10, and a positive review has a score ≥ 7 out of 10. Neutral reviews are not included in the dataset. The dataset is divided into training and test sets. The training set is the same 25,000 labeled reviews.

Conveniently, Keras has a built-in IMDb movie reviews data set that we can use. You can import it as follows:

```
from keras.datasets import imdb
```

You can now load the dataset using the load_data function.

If you look at the data once loaded you will realise that it has been already pre-processed. All words in a review have been mapped to integers.

```
In [15]: print('---review---')
         print(X_train[6])
         print('---label---')
         print(y_train[6])
```

```
---review---
[1, 2, 365, 1234, 5, 1156, 354, 11, 14, 2, 2, 7, 1016, 2, 2, 356, 44, 4, 1349, 500, 746, 5, 200, 4, 4132, 11, 2, 2, 1
117, 1831, 2, 5, 4831, 26, 6, 2, 4183, 17, 369, 37, 215, 1345, 143, 2, 5, 1838, 8, 1974, 15, 36, 119, 257, 85, 52, 48
6, 9, 6, 2, 2, 63, 271, 6, 196, 96, 949, 4121, 4, 2, 7, 4, 2212, 2436, 819, 63, 47, 77, 2, 180, 6, 227, 11, 94, 2494,
2, 13, 423, 4, 168, 7, 4, 22, 5, 89, 665, 71, 270, 56, 5, 13, 197, 12, 161, 2, 99, 76, 23, 2, 7, 419, 665, 40, 91, 85
, 108, 7, 4, 2084, 5, 4773, 81, 55, 52, 1901]
---label---
1
```

These integers represent the words sorted by their frequency. So 4 represents the 4th most used word, 5 the 5th most used word and so on. The integer 1 is reserved for the start marker, the integer 2 for an unknown word and 0 for padding. The label is also an integer (0 for negative, 1 for positive).

If you want to take a look at the reviews yourself and see what people have actually written, you can map the review back to the original words:

```
In [16]: word2id = imdb.get_word_index()
         id2word = {i: word for word, i in word2id.items()}
         print('---review with words---')
         print([id2word.get(i, ' ') for i in X_train[6]])
         print('---label---')
         print(y_train[6])
```

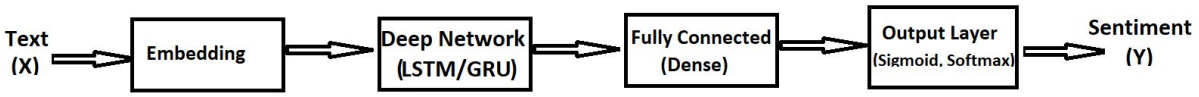
```
---review with words---
['the', 'and', 'full', 'involving', 'to', 'impressive', 'boring', 'this', 'as', 'and', 'and', 'br', 'villain', 'and',
'and', 'need', 'has', 'of', 'costumes', 'b', 'message', 'to', 'may', 'of', 'props', 'this', 'and', 'and', 'concept',
'issue', 'and', 'to', "god's", 'he', 'is', 'and', 'unfolds', 'movie', 'women', 'like', "isn't", 'surely', "i'm", 'and
', 'to', 'toward', 'in', "here's", 'for', 'from', 'did', 'having', 'because', 'very', 'quality', 'it', 'is', 'and', '
and', 'really', 'book', 'is', 'both', 'too', 'worked', 'carl', 'of', 'and', 'br', 'of', 'reviewer', 'closer', 'figure
', 'really', 'there', 'will', 'and', 'things', 'is', 'far', 'this', 'make', 'mistakes', 'and', 'was', "couldn't", 'of
', 'few', 'br', 'of', 'you', 'to', "don't", 'female', 'than', 'place', 'she', 'to', 'was', 'between', 'that', 'nothin
g', 'and', 'movies', 'get', 'are', 'and', 'br', 'yes', 'female', 'just', 'its', 'because', 'many', 'br', 'of', 'overl
y', 'to', 'descent', 'people', 'time', 'very', 'bland']
---label---
1
```

Preprocessing the Data

In order to feed this data into our RNN, all input documents must have the same length. Since the reviews differ heavily in terms of lengths we need to trim each review to its first 500 words. If reviews are shorter than 500 words we will need to pad them with zeros. Keras, offers a set of preprocessing routines that can easily do this for us. In order to pad our reviews we will need to use the pad_sequences function.

Build the Model

We can now build our model. Deep learning text classification model architectures generally consist of the following components connected in sequence:



A word embedding is a class of approaches for representing words and documents using a dense vectors where a vector represents the projection of the word into a continuous vector space.

The position of a word within the vector space is learned from the text and is based on the words that surround the word when it is used. The position of a word in the learned vector space is referred to as its embedding.

Keras offers an embedding layer that can be used for neural networks on text data. It requires that the input data be integer encoded, so that each word is represented by a unique integer.

The embedding layer is trained as a part of the RNN and will learn to map words with similar semantic meanings to similar embedding-vectors. It is initialized with random weights and will learn an embedding for all of the words in the training dataset.

The Embedding layer is defined as the first hidden layer of a network. It must have three arguments:

- input_dim: This is the size of the vocabulary in the text data. For example, if your data is integer encoded to values between 0-5000, then the size of the vocabulary would be 5001 words.

- output_dim: This is the size of the vector space in which words will be embedded. It defines the size of the output vectors from this layer for each word. For example, it could be 32 or 100 or even larger. Test different values for your problem.

- input_length: This is the length of input sequences, as you would define for any input layer of a Keras model. For example, if all of your input documents are comprised of 500 words, this would be 500.

Train the Model

You are now ready to train your model. Remember to compile your model by specifying the loss function and optimizer we want to use while training, as well as any evaluation metrics we'd like to measure.

Once compiled, you can start the training process. Note that there are two important training parameters that we have to specify, namely batch size and number of training epochs. Together with our model architecture these parameters determine the total training time.

Test the Model

After training the model go ahead and test the model. Check the accuracy of your model. If low, check what values you can tweak in order to increase the accuracy.

Predict Something

.

Of course at the end we want to use our model in an application. So we want to use it to create predictions. In order to do so we need to translate our sentence into the corresponding word integers and then pad it to match our data. We can then feed it into our model and see if it thinks we liked or disliked the movie.

Compulsory Task 1

Follow these steps:

- Use the steps outlined above to create a recurrent neural network that is able to classify a movie review as either positive or negative.
- When complete, push this project to GitHub. Include a detailed README file in which you describe your project in detail.

Once you have completed the task in line with the instructions above, click the button below to request your mentor to review your work and provide feedback. If you have any questions while attempting the task, leave your mentor a note on the comments.txt file in your Student Dropbox folder.

Completed the task(s)?

Ask your mentor review your work!

Review work

Things to look out for:

1. Make sure that you have installed and setup all programs correctly. You have setup **Dropbox** correctly if you are reading this, but **Python or Notepad++** may not be installed correctly.
2. If you are not using Windows, please ask your mentor for alternative instructions.

Rate us

Share your thoughts

Hyperion strives to provide internationally-excellent course content that helps you achieve your learning outcomes.

Think the content of this task, or this course as a whole, can be improved or think we've done a good job?

Click here to share your thoughts anonymously.