Mini Project: Using Keras to analyze IMDB Movie Data

Now, you're ready to shine! In this project, we will analyze a dataset from IMDB and use it to predict the sentiment analysis of a review.

Workspace

To open this notebook, you have two options:

- Go to the next page in the classroom (recommended)
- Clone the repo from Github and open the notebook **IMDB_in_Keras.ipynb** in the **imdb_keras** folder. You can either download the repository with `git clone https://github.com/udacity/deep-learning.git`, or download it as an archive file from this link.

Instructions

In this lab, we will preprocess the data for you, and you'll be in charge of building and training the model in Keras.

The dataset

This lab uses a dataset of 25,000 IMDB reviews. Each review, comes with a label. A label of 0 is given to a negative review, and a label of 1 is given to a positive review. The goal of this lab is to create a model that will predict the sentiment of a review, based on the words on it. You can see more information about this dataset in the Keras website.

Now, the input already comes preprocessed for us for convenience. Each review is encoded as a sequence of indexes, corresponding to the words in the review. The words are ordered by frequency, so the integer 1 corresponds to the most frequent word ("the"), the integer 2 to the second most frequent word, etc. By convention, the integer 0 corresponds to unknown words.

Then, the sentence is turned into a vector by simply concatenating these integers. For instance, if the sentence is "To be or not to be." and the indices of the words are as

- "be": 8
- "or": 21
- "not": 3

Then the sentence gets encoded as the vector `[5,8,21,3,5,8]`.

Loading the data

The data comes preloaded in Keras, which means we don't need to open or read any files manually. The command to load it is the following, which will actually split the words into training and testing sets and labels!:

```
from keras.datasets import imdb
(x_train, y_train), (x_test, y_test) = imdb.load_data(path="imdb.npz",
                                                      num_words=None,
                                                      skip_top=0,
                                                      maxlen=None,
                                                      seed=113,
                                                      start_char=1,
                                                      oov_char=2,
                                                      index_from=3)
```

The meaning of all these arguments is here. But in a nutshell, the most important ones are:

- **num_words**: Top most frequent words to consider. This is useful if you don't want to consider very obscure words such as "Ultracrepidarian"
- **skip_top**: Top words to ignore. This is useful if you don't want to consider the most common words. For example, the word "the" would add no information to the review, so we can skip it by setting `skip_top` to 2 or higher.

Pre-processing the data

We first prepare the data by one-hot encoding it into (0,1)-vectors as follows: If, for example, we have 10 words in our vocabulary, and the vector is (4,1,8), we'll turn it into the vector (1,0,0,1,0,0,0,1,0,0).

Building the model

regularization, and good Keras optimizers to do this. A good accuracy to aim for is 85%. Can your model achieve this?

Help

This is a self-assessed lab. If you need any help or want to check your answers, feel free to check out the solutions notebook in the same folder, or by clicking **here**.

NEXT