Using User Reviews to Enrich Social Recommender Systems – Deployment Guide

Link to Project: Using User Reviews to Enrich Social Recommender Systems

1 Initial Setup

The key features of the proposed model were developed using **Python 3.9**, **Keras 2.5.0** and **Gensim 4.0.1**.

1.1 Libraries

Please ensure that all of the necessary libraries have been installed. These include:

- Argparse
- Gensim
- Matplotlib
- Keras
- NLTK
- Numpy
- Pandas
- Pydot
- Sci-kit Learn
- Tensorflow

1.2 Dataset

To ensure that the code executes correct, it is necessary to have the correct dataset files within the required directory. The project utilises three dataset files which can be downloaded via the hyperlinks, users, businesses and reviews. Alternatively, you can download the full yelp dataset here.

1.3 File Extraction

Go to the link provided at the top of the document and download the entire project. Upon completion, extract the files into the desired location. The files that you downloaded in the previous subsection should be placed within this directory. Thus, the directory should have a similar hierarchy to that depicted in Figure I.

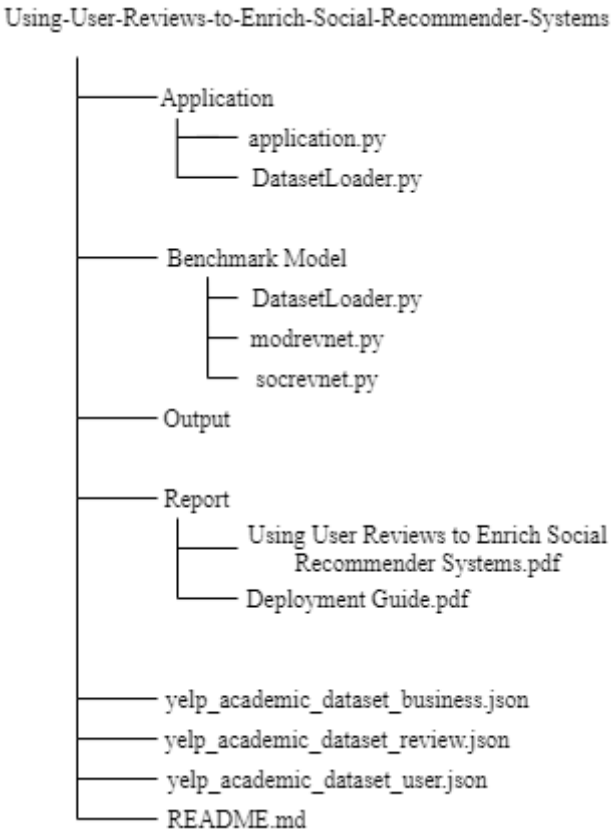

Figure I: Directory Hierarchy

2 Code Execution

There are two ways of running the provided code. The following subsections will illustrate the process that needs to be followed.

2.1 Default Execution

To run the code with the models default configuration, follow the steps below using the command line/terminal.

i. Navigate to the **Application** directory.
ii. Enter "**python application.py**" into the command line/terminal.

2.2 Parameter Tuning

There are a multitude of parameters that can be experimented with when executing the provided code. In this regard, we utilise argparse to extract command line parameters. To see the full list of commands possible, enter to following command into you command line/terminal.

- python application.py –help
  OR
- python application.py -h

Running the above commands will provide you with a list of available parameters for which you may enter values for. These are shown below and summarised in Table I.

Table I: Parameters and their descriptions

Parameters			
application.py [-h] [--data DATA] [--users USERS] [--reviews REVIEWS] [--batch_size BATCH_SIZE] [--neural_epochs NEURAL_EPOCHS] [--validation_split VALIDATION_SPLIT] [--test_split TEST_SPLIT] [--leaky_alpha LEAKY_ALPHA] [--d2v_epochs D2V_EPOCHS] [--d2v_vec_size D2V_VEC_SIZE] [--lr LR] [--drp DRP] [--dm DM] [-f F]			
Parameter Representation	**Description**	**Data Type**	**Default**
--data	A flag that identifies whether to read in the full dataset or not. 0- Read in a portion 1- Read in the full dataset	int	0
--users	Specifies the number of users to read in from the user's dataset file	int	1
--reviews	Specifies the number of reviews to read in from the reviews dataset file	int	500
--batch_size	The batch size used for training the neural network	int	64
--neural_epochs	Specifies the number of epochs to train the neural network for	int	100
--validation_split	The split ratio for the validation set	float	0.05
--test_split	The split ration for the test data	float	0.2
--leaky_alpha	The alpha parameter for the Leaky ReLU activation function used in the neural network	float	0.1
--d2v_epochs	The number of epochs used for training of the doc2vec models	int	1
--d2v_vec_size	Specifies the output vector size from the doc2vec model	int	64

--lr	The learning rate for the neural model	float	0.001
--drp	The dropout rate used in the neural model	float	0.2
--dm	Specifies the training algorithm used to train the doc2vec model. 1 – Distributed Memory 2 – Distributed Bag of Words	int	1

To execute the code with custom parameters, follow the format of,

- python application.py --parameter_1 value_1 … --paramter_n value_n

Sample custom executions are demonstrated in Table II.

Table II: Examples of Execution Commands

Command	Description
python application.py --users 1000 --reviews 1000 --d2v_eopchs 20 --d2v_vec_size 32 --dm 1	Run the social recommender with **1000 users** and **1000 reviews** being read in. Furthermore, train the **doc2vec** models using **20 epochs**, with an output **vector size of 32** whilst training on the **Distributed Memory algorithm**
Python application.py --users 1000 --reviews 10000 --lr 0.05 --d2v_eopchs 20 --d2v_vec_size 256 --dm 2	Run the social recommender with **1000 users** and **10 000 reviews** being read in. In the neural network, use a **learning rate of 0.05**. Furthermore, train the **doc2vec** models using **20 epochs**, with an output **vector size of 256** whilst training on the **Distributed Bag of Words algorithm**

Note to user: The training of the model may cause substantial memory usage in the case that you experiment with the *--users* and *--reviews* hyper parameters. As such, ensure that you have in excess of 8GB of RAM when deviating from the default parameter values.