**Project Report**

Introduction/Background

We chose to develop a movie recommendation network using data from the MovieLens Education and Development “small” dataset1. We chose to use a smaller dataset to reduce the time and memory demands, but a larger sets would give more accurate results. This dataset contains 100,836 ratings by 610 users of 9724 movies. Each rating contains a timestamp of when the rating was given, but we ignored them for this project. It also came with a file linking the movie ID’s with the movie title and genres that each movie falls into. For this project, we focused on the movie ratings as our feature space instead of the movie genres. The data is contained in a pandas DataFrame, with each row representing a single user’s ratings and each column corresponding to a specific movie. The ratings range from 1-5 and are half-integer steps (ie. 1, 1.5, 2, 2.5, etc.). The dataset contains null ratings, since not every user has given a rating for every movie, and we set these ratings to 0. This dataframe was then split into training/testing sets, where 80% of the users were used for training and 20% was used for testing. We implemented a single hidden layer, autoencoder network designed by Soumya Ghosh2, which is based off of a paper by Suvash Sedhain *et al*3. The overall architecture design of this network can be seen in Figure 1 below, which comes from the previously mentioned paper.

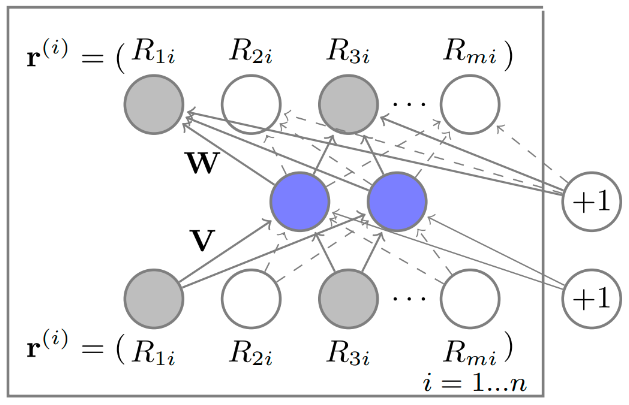


Figure 1: Network design of the autoencoder network used as a movie recommendation model.

This autoencoder network is an unsupervised neural network with a single hidden layer, shown in blue.

Problem Statement

It is beneficial for many companies, such as Netflix or Amazon, to be able to learn the interests of their individual users in order to customize their experience to increase the likelihood of their repeat usage or business. Neural networks have become popular for making user recommendations on large datasets by learning the features that correspond to an individual’s interests or input. For this project, which is similar to the challenge Netflix faces, we have movie ratings given by users and we want to be able to use this information to predict which other movies that user would enjoy watching that they haven’t seen yet. This is very important for Netflix, because if they can reliably recommend movies that their customers enjoy, then their customers will be more likely to continue using their service and pay the monthly fee.

Setup and Running of the Network

The program is run within a Python 3.7 Tensorflow environment. The external libraries necessary to run this program are pandas4, numpy5, tkinter6, and random7. Matplotlib8 was also used to create the plots shown in the Figures section. It is recommended to run this program in a python IDE such as Spyder IDE, which can be accessed by downloading Anaconda9. You can then run the program by selecting the green “Play” button at the top of the IDE. The program will then train the network using the ratings from the MovieLens dataset. Once trained, a window will pop-up that prompts the user to give their own ratings on a select set of movies. Choose a rating from the provided options by clicking in its corresponding box and select “Next Movie” to go to the next movie. If you have not seen the movie and don’t wish to give a rating, select “Next Movie” without selecting a box. Once you have rated as many movies as you’d like, select “Finished Rating” and a new window will open displaying the top five movies that the network has recommended based on your rating input and the learned features from training.

Results/Analysis

Conclusion

Figures

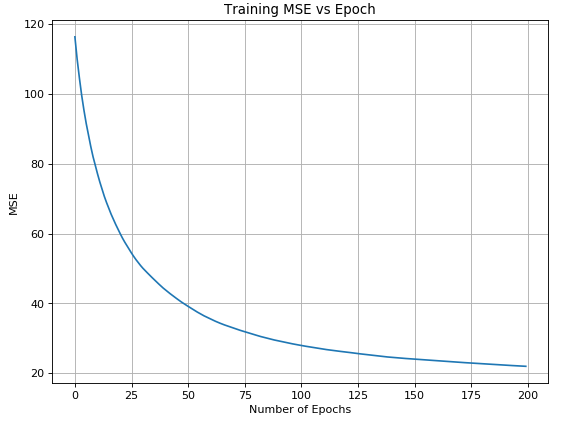


Figure 2: Plot showing the decrease in the Mean Squared Error during training vs. the number of epochs. This was ran with 256 hidden layer neurons, with a 0.1 learning rate.

References

1. GroupLens - MovieLens Dataset  
   <https://grouplens.org/datasets/movielens/>  
   [Accessed Dec. 12, 2018]
2. Soumya Ghosh, *Recommender System on the MovieLens dataset using an Autoencoder and Tensorflow in Python*, Medium.com, March 17, 2018  
   <https://medium.com/@connectwithghosh/recommender-system-on-the-movielens-using-an-autoencoder-using-tensorflow-in-python-f13d3e8d600d>  
   [Accessed Dec. 12, 2018]
3. Suvash Sedhain *et al*., *AutoRec: Autoencoders Meet Collaborative Filtering*, Australian National University  
   <http://users.cecs.anu.edu.au/~akmenon/papers/autorec/autorec-paper.pdf>  
   [Accessed Dec. 12, 2018]
4. Python Data Analysis Library (Pandas)  
   <https://pandas.pydata.org/>
5. Numpy Library, Scipy  
   <http://www.numpy.org/>
6. Tkinter Library, Python  
   <https://docs.python.org/3/library/tkinter.html>
7. Random Library, Python  
   <https://docs.python.org/2/library/random.html>
8. Matplotlib Library  
   <https://matplotlib.org/api/_as_gen/matplotlib.pyplot.plot.html>
9. Anaconda – Python 3.7  
   <https://www.anaconda.com/download/>