IDS576: Assignment 2

Due date: Mar 13 (11.58 PM CT)

Turn in solutions as pdf(s) and ipynb (Python 2 preferred) file(s) on Blackboard.

Note: Answer the following questions concisely, in complete sentences and with full clarity. If in doubt, ask classmates using Slack. Across group collaboration is not allowed. Cite all your sources (see the Syllabus for these and other pointers).

1 Embeddings (25pt)

Instead of embedding words, we will embed movies. In particular, if we can embed movies, then similar movies will be close to each other and can be recommended. This line of reasoning is analogous to the *distributional hypothesis of word meanings* (see https://en.wikipedia.org/wiki/Distributional_semantics). For words, this roughly translates to words that appear in similar sentences should have similar vector representations. For movies, vectors for two movies should be similar if they are watched by similar people.

Let the total number of movies be M. Let $X_{i,j}$ be the number of users that liked both movies i and j. We want to obtain vectors $v_1, ..., v_i, ..., v_j, ..., v_M$ for all movies such that we minimize the cost $c(v_1, ..., v_M) = \sum_{i=1}^{M} \sum_{j=1}^{M} \mathbf{1}_{[i \neq j]} (v_i^T v_j - X_{i,j})^2$. Here $\mathbf{1}_{[i \neq j]}$ is a function that is 0 when $i = j$ and 1 otherwise.

1. Compute data $X_{i,j}$ from the attached csv files [1]. In movieratings.csv, we have 943 users rate 1682 movies generating 100000 observations. Each row in the csv file is movie_id, user_id and rating (1 implies user likes and 0 implies user dislikes). File movies.csv maps movie_ids to the actual names.

2. Optimize function $c(v_1, ..., v_M)$ over $v_1, ..., v_M$ using gradient descent. Do this for two different starting parameters: (a) when all the vectors are zeros, and (b) when each coordinate of each vector is i.i.d random uniform between $[-0.7, 0.7]$. Plot the loss as a function of iteration for both settings.

3. Recommend top 10 movies (not vectors or indices but movie names) given movie 'Aladdin'. Describe your recommendation strategy.

4. Recommend top 10 movies given movies 'Toy Story' and 'Home Alone'. Describe your recommendation strategy.

2 RNNs (25pt)

Note: This problem is quite open-ended, so make any assumptions as necessary.

1. Pick a text corpus or use the provided one (text_corpus.txt[2]).

[1] courtesy Arora and Hazan, COS402.

[2] From Andrej Karpathy's char-rnn repo https://github.com/karpathy/char-rnn

2. Follow the instructions from this page to set up a twitter bot (up to reading tweets from a file).

3. Train an RNN (e.g., LSTM) based character level language model (say using Keras/Tensorflow) from the corpus.

4. Output the character strings (≤ 140) generated by the RNN model to the twitter bot and make it tweet 20 times (1 per 6 minutes).

5. Remove your twitter credentials from the python code and submit your implementation and the name of the twitter bot you deployed.

6. Also describe the model and implementation details. Show training performance as a function of training iterations.

7. (Bonus) Train an n-gram character level language model and repeat the above steps.