Movie Rating Prediction using RBM

Dhruv Khattar - 201402087

Pinkesh Badjatiya - 201402002

Siddhartha Gairola - 201402068

Introduction

- The existing approaches to collaborative filtering are unable to handle huge data sets. So we use a class of 2 layer undirected graphical model called Restricted Boltzmann Machines to model tabular data - which in our case is Movie Ratings given by users. We can use the data sets available on Netflix or IMDB.
- From the above we aim to predict the ratings for the movies which have not yet been given a rating by the users.

Dataset

- Netflix dataset from 1998 to 2005
- 100 million ratings from 480 thousand randomly-chosen, anonymous Netflix customers over 17,000 movies. The ratings are on a scale from 1 to 5. Each user has an integral userid.
- 3 types of data
    - Training data: 100M ratings from 480k users randomly chosen on 18k movies.
    - Validation data: Containing 1.4M ratings. **[NOT AVAILABLE]**
    - Probe Data(Test data): Has 2.8M user/movie pairs without any rating.
- Source:https://web.archive.org/web/20090925184737/http://archive.ics.uci.edu/ml/datasets/Netflix+Prize

Method

1. A low dimensional feature vector is assigned to each user and a low dimensional feature vector to each movie so that the rating each user assigns to each movie is modeled by the scalar-product of the 2 feature vectors.
2. This means that the N × M matrix of ratings that N users assign to M movies is modeled by the matrix X which is the product of an N × C matrix U whose rows are the user feature vectors and a C × M matrix V whose columns are the movie feature vectors. The rank of X is C the number of features assigned to each user or movie.
3. But in case of lot of ratings missing, we model a different RBM for each user. Every RBM has the same number of hidden units, but an RBM only has visible softmax units for the movies rated by that user, so an RBM has few connections if that user rated few movies. Each RBM only has a single training case, but all of the corresponding weights and biases are tied together, so if two users have rated the same movie, their two RBM's must use the same weights between the softmax visible unit for that movie and the hidden units.

Model

1. The visible layer consists of x units where x is the number of movies a particular user has rated.
2. Each unit in the visible layer is a vector of length 5 (since ratings are from 1 to 5), and the ith index is one corresponding to the rating the user has given, the rest are zeros.
3. The hidden layer consist of 100 units which are binary.
4. The activation function we have used is the sigmoid function both for forward propagation and backward propagation.

Training & Learning

- 2 cycles - Forward propagation and Backward propagation.
- Weights initially have been assigned randomly.
- Forwards propagation - find the positive associations after finding the hidden units.
- Backward propagation - from the hidden units found in forward propagation we find the visible units and then again do forward propagation to find the negative associations.
- The difference between the positive and the negative associations gives us Δw the value with which we change the current weights for a user in the RBM.

Forward Propagation

$$p(h_j = 1|\mathbf{V}) = \sigma(b_j + \sum_{i=1}^{m}\sum_{k=1}^{K} v_i^k W_{ij}^k)$$

Backward Propagation

$$p(v_i^k = 1|\mathbf{h}) = \frac{\exp\left(b_i^k + \sum_{j=1}^{F} h_j W_{ij}^k\right)}{\sum_{l=1}^{K} \exp\left(b_i^l + \sum_{j=1}^{F} h_j W_{ij}^l\right)}$$

Contrastive Divergence

To avoid computing **<·>** model, we follow an approximation to the gradient of a different objective function called CD.

Expectation **<.>**T represents a distribution of samples from running the Gibbs sampler. Instead, we have used a random sampler.

$$\Delta W_{ij}^k = \epsilon(<v_i^k h_j>_{data} - <v_i^k h_j>_T)$$

Making Predictions:

We compute the probabilities for each rating from 1-5.

The maximum probability from the calculated values gives us the rating which a particular user would give that movie.

$$p(v_q^k = 1|\mathbf{V}) \propto \sum_{h_1,\ldots,h_p} \exp(-E(v_q^k, \mathbf{V}, \mathbf{h}))$$

$$\propto \Gamma_q^k \prod_{j=1}^{F} \sum_{h_j \in \{0,1\}} \exp\left(\sum_{il} v_i^l h_j W_{ij}^l + v_q^k h_j W_{qj}^k + h_j b_j\right)$$

$$= \Gamma_q^k \prod_{j=1}^{F} \left(1 + \exp\left(\sum_{il} v_i^l W_{ij}^l + v_q^k W_{qj}^k + b_j\right)\right)$$

$$\Gamma_q^k = \exp\left(v_q^k b_q^k\right).$$

Results

- RMSE as obtained by our model is ~0.96 with only 1000 users and 25 iterations while the paper reports **~0.91** with about 2M users.