Google's Multitask Ranking System

An overview of Google's Ranking System and discussion of methods

Tech Review - CS 410 - Alejandro Rocas (Paper Source)

Introduction

Google researchers came out with a new paper on a new neural network architecture that can improve recommendations for a YouTube user.

A typical recommendation system works in two stages: First, it generates the possible candidates that can be shown to the user. This step looks at the current video being watched by the user along with user history, user details, etc. For the first step, the YouTube system uses multiple candidate generation algorithms, each of which captures one aspect of similarity between query video and candidate video. For example, one algorithm generates candidates by matching topics of query video. Another algorithm retrieves candidate videos based on how often the video has been watched together with the query video. Other aspects taken into account include user history, and context to name a few. This step results in a few hundred candidate videos which now need to be ranked. In the second stage it ranks these candidates. The paper doesn't give details about the neural network architecture used for the candidate generation step and chooses to focus on the second stage, how to rank the candidates from the first stage.

The Problem Statement

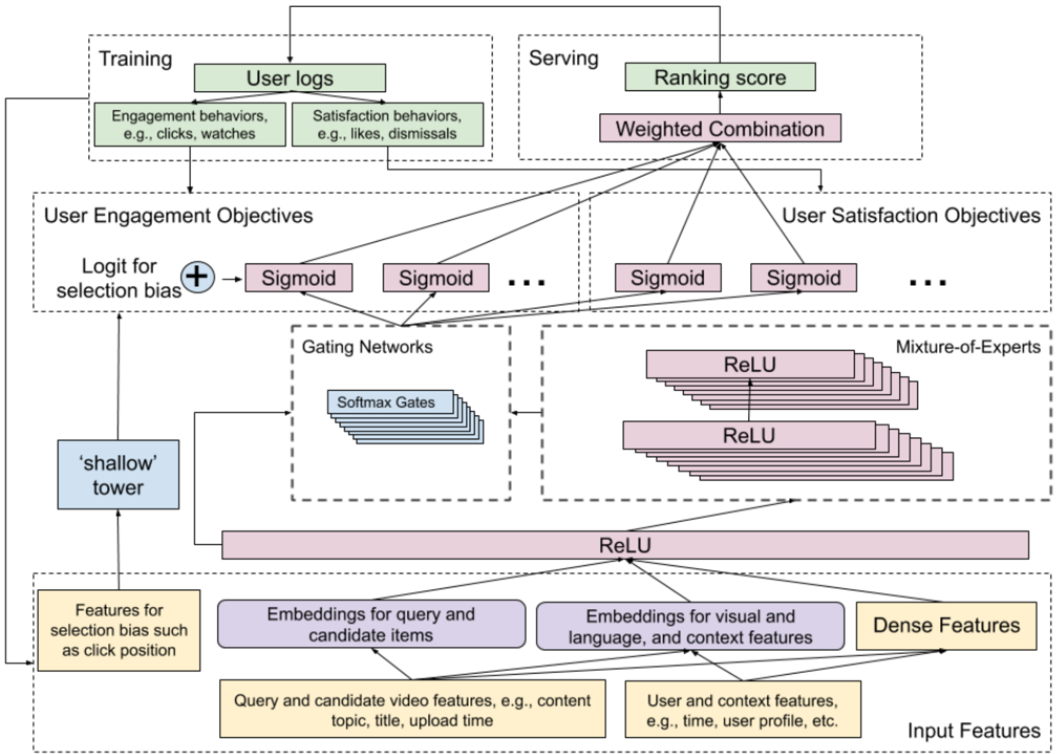
As mentioned earlier, for YouTube, the overall aim of a recommendation system is to take into account the video which a user is currently watching and recommend the next video that the user might watch and enjoy. In the case with google, there are two main problems:
1. Multiple Objective Functions: There are several different and conflicting objective functions that need to be optimized while solving the problem. The paper divides these objectives into two groups:

- Engagement Objectives: These can be measured using data on clicks, the degree of engagement of the user while watching the recommended video.
- b) Satisfaction Objective: These can be measured by data in likes, shares, comments, rating. Both of these objectives contain binary classification tasks and regression tasks.

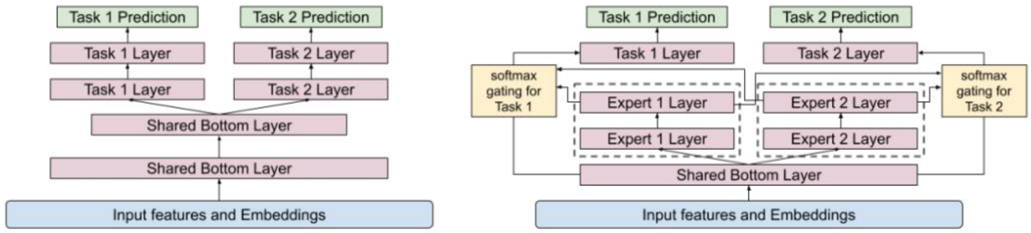
2. Removal of Implicit Bias: The data used for training the model contains some implicit bias. This is because a user historically might have clicked and watched a video simply because it was being ranked high, not because it was the one that the user liked the most. If the model is trained using such data, it will produce biased non-optimal recommendations which the user might not like.

The Solution

With the above objectives in mind, the researchers have designed the below network architecture:



The model starts with multiple input features and embeddings which are first fed into a shared hidden layer. This is done because supplying the input features directly to the next layer (called MMoE layer) significantly increases the cost of training. MMoE is essentially a combination of Multi-Layer Perceptrons followed by ReLU activations (The rectified linear activation function or ReLU for short is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero. Each of the experts in the MMoE layer tries to learn a different feature of the input. The output of the MMoE layer is then fed into a Gating Network. The output of these Gating Networks and that of the shared hidden layer are then fed into the various objective functions i.e., engagement as well as satisfaction. Each objective function is represented by a sigmoid activation function. During training, each of these objectives looks at each of the experts and chooses one or more out of these experts that are relevant for deciding that objective function. This takes care of the problem of the multiple conflicting objective functions.



(a) Shared-Bottom Model with shared bottom hidden layers and separate towers for two tasks.

(b) Multi-gate Mixture-of-Expert Model with one shared bottom layer and separate hidden layers for two tasks.

Moving on to handling bias, the ideal data to train a recommendation system would be an explicit feedback data from the user about whether they like the video recommendation or not. Since such data is not always available or is expensive to collect, implicit feedback data is used for training. Implicit feedback means that if a user clicks on the recommendation, it is believed that the user likes the recommendation. However, this may not be always true. A user might be clicking on a video just because it appears on top of the list of recommendations. Using this data for training might not be ideal as there is a bias in this data. This bias needs to be removed while training the model. To address this, a shallow tower is introduced into the model architecture. The shallow tower is trained using features that contribute to the bias like position of the recommendation and attempts to predict whether there is a bias component involved in the current instance. The selection bias output is also fed an input to the engagement objectives to make the network learn to remove these biases.

Conclusion

The proposed model architecture is similar to the Wide & Deep model architecture. People memorize everyday events and then generalize those learning to apply to things we haven’t seen before. Similarly, a Wide and Deep Architecture jointly trains a wide linear model (memorization) alongside a deep neural network (generalization) which can help the machine think similar to people. In this case, the part is the MMoE portion while the wide part is the shallow tower.