Introduction

In week 6 of this course the concept of “recommender systems” was introduced. We were taught two broad methodologies in the recommendation space: content-based (item similarity) and collaborative filtering (user similarity). The information that was covered in week 6 [2] gives us a very general understanding of the difficulty of recommendation but does not necessarily provide the exact implementation. In the following sections, I will be presenting a framework presented by Google which utilizes deep learning to implement a recommender system [1].

Wide and deep learning technique

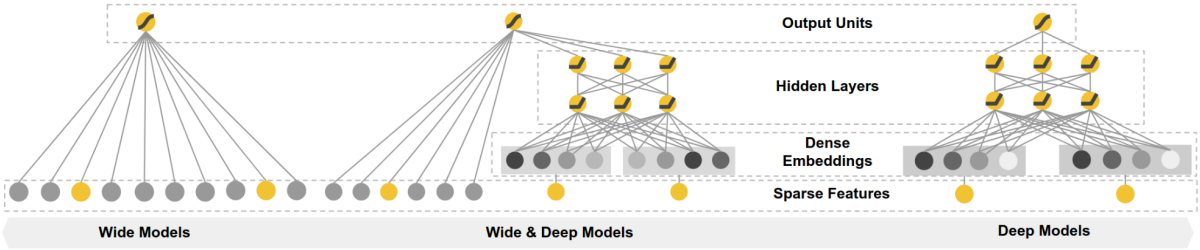

Figure 1: Wide and deep models from the paper [1]

Memory-based approach

In lesson 6.8 we learned about the memory-based approach in recommendation which is the idea of storing all user information and at the time of recommendation for a specific user we try to leverage the recommendations of similar users.

Wide models like logistic regressions are popular for memory-based approaches because they can use cross-product transformations to create interaction terms for categorical variables.

Since memorization tries to exploit the information from the historical data i.e on items that already have interactions, it struggles with cold-start and diversity of recommendations [2].

Generalization approach

The generalization approach's goal is to improve the diversity of recommendations by solving the cold-start problem. Recommender systems balance memorization and generalization to provide richer recommendations.

Embedding-based models are a popular approach in solving generalization. Embedding-based models work by learning a low-dimensional dense embedding vector for each user, item feature. This is very similar to the vector-space-model from lessons 2.X [2].

The issue with this approach is over-generalization. Over-generalization is caused by mapping from sparse features to dense embeddings. So in the case where a user has a very specific niche one would expect that there should be very few interactions between this user and any other items but the dense embedding will create spurious correlations between this user and items.

Combining memorization and generalization

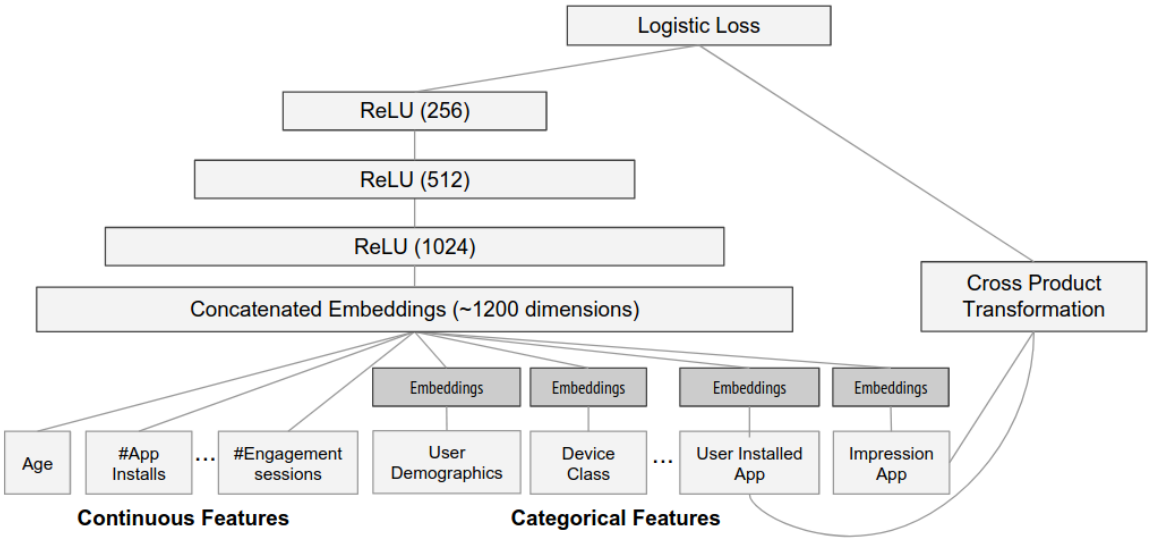

Figure 2: model architecture to leverage both memorization and generalization taken from [1]

Since logistic regression handles memorization and embedding-based neural network models can handle generalization, it is natural to then combine the two different models to achieve both memorization and generalization. From Figure [2] the Cross Product Transformation block represents the logistic regression where the feature are

engineered from categorical characteristics. The other blocks represent the deep neural network which creates embeddings from the categorical variables and appends them to the continuous variables to create a single large embedding. The model is jointly trained and the training and deployment architecture is beyond the scope of this review.

Conclusion

Memorization and generalization are two objectives that need to be balanced to provide richer recommendations. The paper by Cheng et al. [1] presents a machine learning based model which can balance the two objectives by leveraging a wide logistic regression to handle memorization and a deep neural network to handle generalization. This general framework can be extended to many different recommender system applications.

References

[1] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. In RecSys, 2016

[2] Zhai *Text Information Systems* [Mooc]. Coursera. https://www.coursera.org/learn/cs-410/