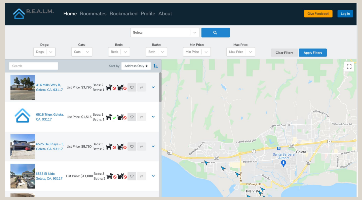
Deep Learning Recommendation Systems for Property Management Software

Alyssa Keehan, Jordan Tran, Joshua Harasaki, Philip Carey & Romina Fareghbal
Mentor: Erika McPhillips
Sponsors: Shyr-Shea Chang and Soeren Thust

Introduction / Background

We created a recommender system that predicts users interests and recommends properties to users based on previous user interactions and property features. Recommender systems are among the most powerful machine learning systems that are used to help users identify items they may be interested in and also helps companies increase interaction and sales. Our sponsor is AppFolio, a tech company based in Santa Barbara that provides innovative software, services, and data analytics to the real estate industry. Building this model will allow AppFolio to analyze the results of our recommender system to assess how interaction data can be used to better their products. We explored multiple models including the Bilateral Variational Autoencoder (BiVAE), TriVae, Visual Bayesian Personalized Ranking (VBPR), and Deep Cross Network (DCN) in an effort to achieve the highest prediction accuracy using 16 property features from our given dataset and images of the properties. Ultimately, we want to integrate our model into the Santa Barbara property website (shown below), which was built by the AppFolio Sponsored CS Capstone team.



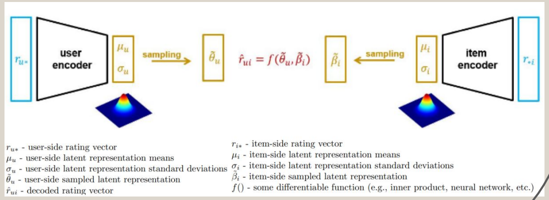
Data

We got our data from LISA, Appfolio Award-Winning AI Leasing Assistant Chatbot, which is designed for property companies to streamline the process of receiving inquiries about properties from users. Our data is divided into three components:user ratings, features of the individual properties, and images. It consists of over four million users, roughly sixteen thousand properties with their corresponding images, and five million user interactions. The only problem we had with missing data was in our property features dataset because there were many null values. To bypass this problem, we just eliminated all the observations with missing values because our dataset was still already large enough that we can implement any model on it.

Methodology

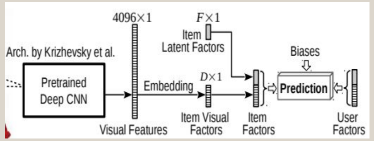
Bilateral Variational Autoencoder

The Bilateral Variational Autoencoder (BiVAE) is a collaborative filtering approach that consists of a seperate variational autoencoder for both the property ratings and user ratings. The output from the two autoencoders are then combined to produce the respective user ratings. These produced ratings are then sorted and the highest rated properties are used as recommendations. A key component of the BiVAE are the constrained adaptive priors used for the sampling distributions. They allow us to incorporate additional features beyond interactions and strengthen our recommendations.


r_{u*}
user encoder
μ_u
σ_u
sampling
$\tilde{\theta}_u$
$\hat{r}_{ui} = f(\tilde{\theta}_u, \tilde{\beta}_i)$
$\tilde{\beta}_i$
sampling
μ_i
σ_i
item encoder
r_{*i}
r_{u*} - user-side rating vector
μ_u - user-side latent representation means
σ_u - user-side latent representation standard deviations
$\tilde{\theta}_u$ - user-side sampled latent representation
$\hat{r}_{ui}$ - decoded rating vector
r_{i*} - item-side rating vector
μ_i - item-side latent representation means
σ_i - item-side latent representation standard deviations
$\tilde{\beta}_i$ - item-side sampled latent representation
$f()$ - some differentiable function (e.g., inner product, neural network, etc.)

$$p(\boldsymbol{\theta}_u) = \mathcal{N}(\mathrm{T}\boldsymbol{\xi}_u, \mathrm{I}), \quad p(\boldsymbol{\beta}_i) = \mathcal{N}(\mathrm{B}\boldsymbol{\lambda}_i, \mathrm{I}),$$

$T \sim N(\mu_u, \sigma_u)$ - User side latent representation
ξ_u - User features latent representation
$B \sim N(\mu_i, \sigma_i)$ - Item side latent representation
λ_i - Item features latent representation



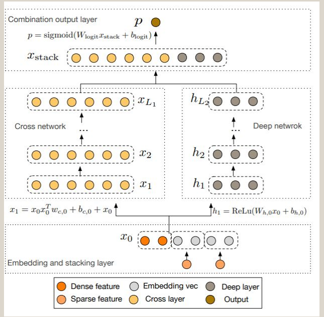
Visual Bayesian Personal Ranking

When first working with images, we looked at the Bayesian Personalized Ranking (BPR) Framework. BPR is a factorization method that uses a pairwise loss function that maximizes the distance between latent factors of each user's interacted properties and non-interacted properties. The BPR framework is flexible and allows us to easily append visual features to the item factors. Therefore, BPR allows us to easily compare performance between BPR with visual image features incorporated (VBPR) and the standard BPR. We first used the pre-trained convolutional network, VGG16, to produce a dense vector of visual features and added it to the item factors. We saw significant performance improvements over the standard BPR and the experiment helped us conclude the usefulness of visual features in recommender systems. We then used a more modern pre-trained convolutional network, EfficientNet, and saw even better performance. VBPR with EfficientNet is currently our best performing implicit feedback model.

Deep Cross Network

Our final model we explored was the Deep Cross Network (DCN). The DCN is a supervised learning approach that is trained on explicit feedback. Explicit feedback would be whether they denied a property viewing or inquired about it. The key part of DCN are synthetic features formed by multiplying (crossing) two or more features which can provide predictive abilities beyond what those features can provide individually. It has been shown learning effective feature crosses is a powerful tool for building recommender systems. DCN utilizes embedding layers and a cross network that learns feature crosses. DCN also allows us to add additional architectures in parallel to the cross network and concatenate their outputs to be fed into the logit layer . We add a fully connected deep neural network that further learns non-linearity of our features and also a pre-trained convolutional neural network, EfficientNet, to incorporate our visual features.

Our final DCN model had a log loss of .4438 and an AUC of .8048. Comparisons between implicit and explicit models is difficult due to the treatment of non-interactions. We have been working on using the BPR loss function in conjunction with DCN to build the model on implicit feedback. We also plan to perform inference using both types of models to see the quality of recommendations. This will allow us to compare the effectiveness between implicit and explicit models.


Combination output layer
p
$p = \mathrm{sigmoid}(W_{\mathrm{logit}} x_{\mathrm{stack}} + b_{\mathrm{logit}})$
x_{stack}
x_{L_1}
Cross network
x_2
x_1
h_{L_2}
Deep netwrok
h_2
h_1
$x_1 = x_0 x_0^T w_{c,0} + b_{c,0} + x_0$
$h_1 = \mathrm{ReLu}(W_{h,0} x_0 + b_{h,0})$
x_0
Embedding and stacking layer
Dense feature
Sparse feature
Embedding vec
Cross layer
Deep layer
Output

Final Results

We've yielded our best results to date from the VBPR model in conjunction with the EfficientNet image preprocessor. The model has potential to include modality for both item features and image features, though we've only implemented the modality for images at this point. However, it still outperforms any other model that we have experimented with.

```
TEST:
...
     |    AUC |    MAP | NDCG@100 | Precision@10 | Precision@100 | Recall@10 | Recall@100 | Train (s) | Test (s)
---- + ------ + ------ + -------- + ------------ + ------------- + --------- + ---------- + --------- + --------
VBPR | 0.9873 | 0.2200 |   0.3522 |       0.0558 |        0.0092 |    0.5192 |     0.8440 | 8798.1470 | 369.6784
```

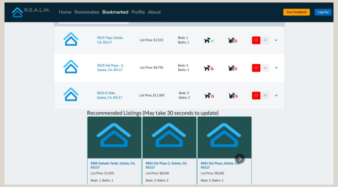
We are most focused on the results of Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG). Both of these metrics are "ranking-aware," meaning we take into account the actual ranking position of relevant items and implement a penalization/reward system based on that position. Additionally, we were able to improve our results with VBPR by increasing the latent dimensions of the users and images and decreasing the batch size during model training.

Conclusions

For this project, we received user interaction data from Appfolio's AI Leasing Assistant, LISA, to build a recommendation system on a website provided by Appfolio's Computer Science Capstone team. Our first couple of weeks in the capstone project consisted of familiarizing ourselves with the data and the recommendations Cornac has to offer. While we were certain the Bilateral Variational Autoencoder would be our strongest model at the end of Winter Quarter, it proved to be weaker than other baselines once we added complexity (i.e. more features and images) to the system. For implementing images into the model, we got our best performing model when preprocessing the images using Efficient Net and using Visual Bayesian Personal Ranking.

Looking back on the project, we had a lot of challenges mainly dealing with the computational power and complexity of this project. Due to our project being based on real-world data and being implemented into a working website, there were logistics which we had to overcome (e.g., version control, implementation of niche packages, and computation time). Since our recommendation package Cornac isn't too widely known, we had very little resources to look at when trying to understand the model's complexity.

For future work, we would like to expand the geographical regions which our model can be applied to since we currently only have data for the Santa Barbara region. In addition, the images we used only contained the front view of the outside of each property. We think it would be valuable to include images of other angles of the house as well as the inside in our model.



Access the website here!

References / Acknowledgements

We'd like to thank the Appfolio CS Capstone Team for collaborating with us to build our recommender system on their website. We would also like to thank our sponsors Shyr-Shea Chang and Soeren Thust and our mentor Erika McPhillips for guiding us throughout the project.