Next Basket Recommendation using Attention

Kriti Aggarwal (A53214465)
Sudhanshu Bahety (A53209213)
Digvijay Karamchandani (A53220055)

Abstract— **Predicting the items in the next basket for a user is a challenging task. The state of the art methods include 'Markov Chains' which focus on predicting the next basket only using the last basket. While, Yu et al proposes a novel model called DREAM(Dynamic REcurrent bAsed Model)[2] which learns both the dynamic representation of a user and the global sequential features of the user baskets. The report shows that the model architecture helps to capture the user's dynamic interests at different times while also maintaining the interactions of all baskets of the user over time. In this report, we present two extensions to the original model. First, we experimented by applying the DREAM model on a Instacart[1] dataset. Second, we added attention on the last few hidden representations of the user to capture the most important latent user representations.**

I. INTRODUCTION

The workflow in most of e-commerce websites consists of the user purchasing different set of items at different times. Hence, predicting the next set of items user would order directly impacts the sales of these websites and also enhace user experience. This is the reason, this problem has received a lot of traction lately.

In this assignment, we try to solve this problem using recurrent neural networks trained end to end to generate the embeddings of user and items. We based our model on the DREAM(Dynamic REcurrent bAsed Model) model proposed by Yu et al. We first analyzed and reproduced the results of the baseline DREAM model.

We then performed two extensions to the original model. One, we changed the dataset used to train the model to Instacart online grocery dataset. Second, we added attention on the last few hidden representations of the user to capture the most important latent user representations.

II. EXPLORATORY ANALYSIS

A. Dataset Description

For the task of next basket prediction, the authors of the DREAM model use two real-world datasets Ta-Feng and T-mall. The Ta-Feng dataset contained multiple purchased baskets from a grocery store where each basket contains multiple items. This dataset contains 817,741 transactions, belonging to 32,266 users and 23,812 items. The T-Mall dataset is a small e-commerce dataset by Alibaba group containing 4,298 transactions of 884 users and 9,531 brands.

Both the Ta-Feng and T-Mall dataset were unavailable to us. Hence, we tried to find the most appropriate and similar dataset useful for our case. We experimented with several datasets and finally used Instacart Online Grocery Shopping Dataset. This dataset consists of over 3.4 million anonymized grocery orders from more than 206,000 users. Each user has on average 16 baskets where each basket contains 4 to 100 items. We performed some preliminary analysis on the dataset which is summarized below.

B. Data Preprocessing

Due to resource constraints, we sampled 0.52 million orders for 32,000 users. We used one hot encoding to initialize the items with two extra one-hot encoding vectors. One for out of vocabulary item and other for padding. For every batch, the number of items in a basket can be different. To alleviate this issue, we performed padding of the items in the basket according to the maximum number of items in any basket within a batch. We took only those user's data who had purchased at least 4 times or more, which could be helpful for the recurrent network to learn dynamic representation of the user.

The dataset was split into training and testing with 80% of the user's data used for training and the rest of the user's data used for testing. The user's data that was used during testing phase was not seen during the time of the training as we assumed that various users' would tend to exhibit similar pattern while shopping and our model would be able to capture the latent representation of user's shopping pattern as they shop.

C. Analysis

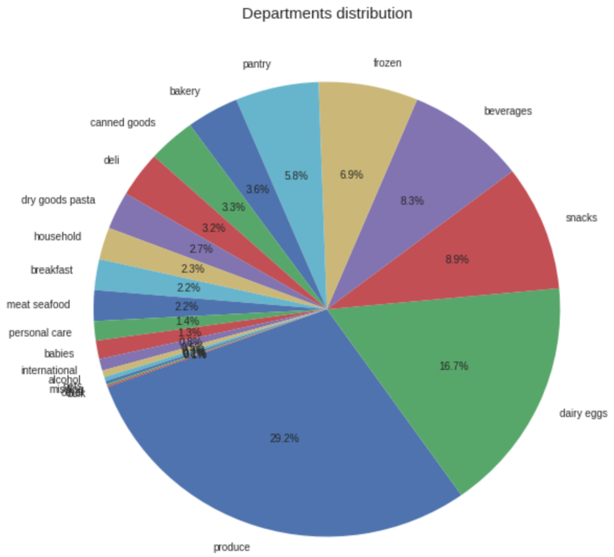
Figure 1 is a pie chart that provides with distribution of products being purchased by the users on the Instacart. The most frequent product purchased by the users are produce followed by dairy eggs. Produce is close to 29.2% of the user's order whereas dairy eggs is close to 16.7%.

Figure 2 shows the distribution number of purchases v/s number of products per order. It's a right tailed distribution with the maximum value at 5 which is around 9000.

Figure 3 shows the number of users v/s the maximum number of orders. We see that there are no users in order in our dataset with less than 4 orders. For simple visualization, we cap the maximum number of orders to 100.

Figure 4 shows the average purchase cycle pattern for the users. We can see that most of the user periodically purchases after a week or after a month. Although it's surprising to see that monthly purchases are most frequent which suggests that generally users tend to use instacart once a month for their orders.

Fig. 1: Pie Chart showing product distribution based on purchase frequency



III. Related Work

Two of the most popular existing approaches for recommender task are mainly based on collaborative filtering and markov chains. The general method to perform collaborative filtering is matrix factorization(MF)[3].

Fig. 2: Number of purchases v/s number of products per order

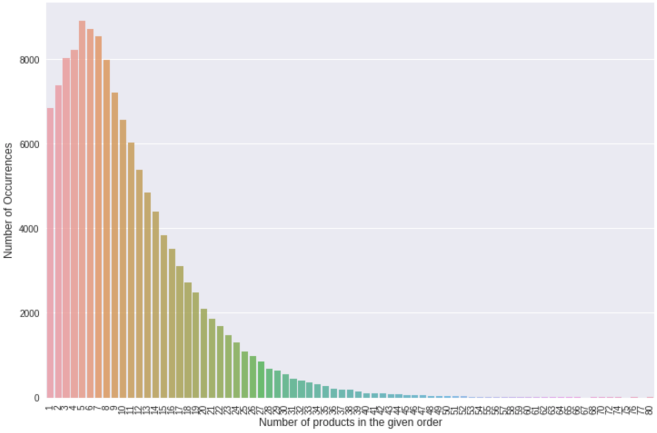
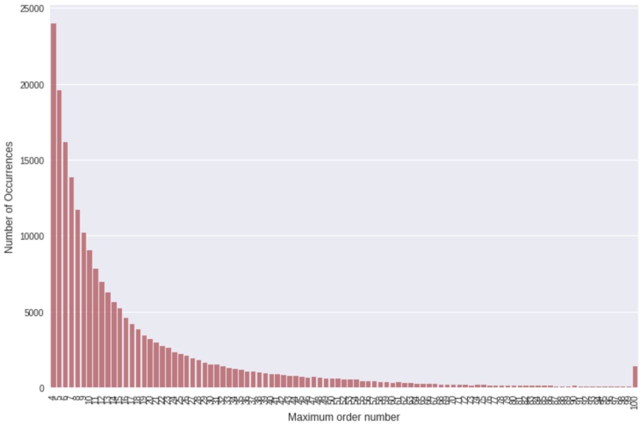


Fig. 3: Number of users vs Maximum number of orders

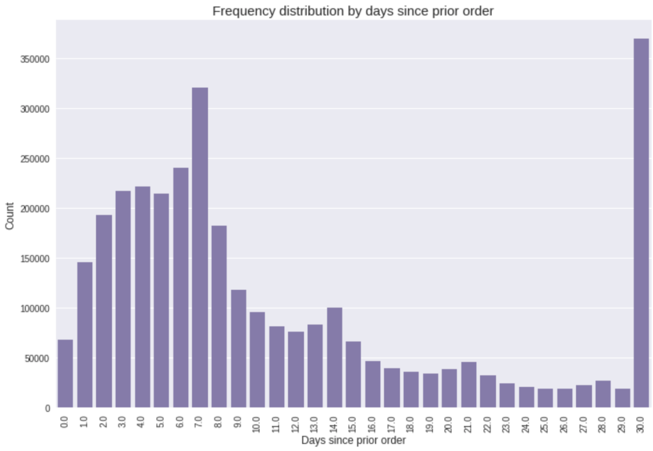


Through MF, the model learns the latent representation of the users and items. Collaborative filtering captures the general user interests but fail to consider the sequential features from historical transactions. For modeling sequential dependencies in the data, Markov Chain is used which predict the next purchase from historical transactions.

Steffen et al's paper uses Factorizing Personalized Markov Chains(FPMC)[4] which models sequential behaviors between adjacent baskets and the general interests of the user.However, FPMC only models linear interaction between these multiple factors. Hierarchical Representation Model(HRM) partially solves the problem of summarizing multiple interacting factors by using max pooling operation[5].

But both of these methods fail to model the depen-

Fig. 4: Number of users vs days since last purchased



dencies on the user behavior due to past baskets since both methods only model the local sequential behavior of the basket between every 2 adjacent baskets.

Shengxian et al's paper uses neural networks to learn the next basket for a user [6]. They modeled the problem by learning the embedding of the baskets and predicting the next basket using dependencies between next and nearest items, while ignoring the user's general interest. This technique also fails to model long term user interests.

Hence, we decided to use recurrent neural networks in this assignment to solve this problem. The reason being that the recurrent neural networks have been successfully applied to sequential prediction problems such as language model, image captioning etc.

IV. Models Used

In order to understand our models performance, first implemented and evaluated the baseline DREAM model by Yu et al. We then modified the original model by adding attention at each input level. In this section, we first define the baseline model and then we go on to defining the attention model that we built over it's top.

A. Baseline

The general framework of DREAM is illustrated in Figure 5. Each input instance in DREAM is a sequence of baskets corresponding to a particular user's shopping. We represent this as $B^u_{t_i} = \{n^u_{t_i,j} \in \mathbb{R}^d | j = 1,2,...,|B^u_{t_i}|\}$. $n^u_{t_i,j}$ is the latent representation of the $j-th$ item in basket $B^u_{t_i}$ which is learnt by the network itself and $|B^u_{t_i}|$ means the number of items in basket $B^u_{t_i}$. We generate the latent vector representation $\mathbf{b}^u_{t_i}$ for a basket $B^u_{t_i}$ by aggregating representation vectors of these items. In this work, we adopt *max pooling* to get the basket representation.

For the *max pooling* operation, we aggregate a group of latent representation of item vector through taking the maximum value of every dimension among all those vectors. Then each dimension of $\mathbf{b}^u_{t_i}$ is formulated as

$$b^u_{t_i,k} = max(n^u_{t_i,1,k}, n^u_{t_i,2,k}, ...)$$

where $b^u_{t_i,k}$ is the k-th dimension of a basket-representing vector $b^u_{t_i}, n^u_{t_i,j,k}$ means the value of k-th dimension of the vector representation of the j-th item $(n^u_{t_i,j})$ in basket $B^u_{t_i}$. We use *max-pooling* operation to add non-linearity in the model.

As is shown in Figure 5, the vector representation of a hidden layer $h^u_{t_i}$ is the dynamic representation of user u at time t_i. The matrix $\mathbf{R}$ encompasses the transitioning of the representation of the user between successive purchases and how the dynamic representation $\mathbf{h}^u_{t_{i-1}}$ and $\mathbf{h}^u_{t_i}$ interacts. $\mathbf{X}$ is a transition matrix between latent vector representations of baskets. We can write the vector representation of the hidden layer as:

$$\mathbf{h}^u_{t_i} = f(\mathbf{X}\mathbf{b}^u_{t_i} + \mathbf{R}\mathbf{h}^u_{t_{i-1}})$$

where $\mathbf{b}^u_{t_i}$ is a latent vector representation of the users basket at time t_i, and $\mathbf{h}^u_{t_{i-1}}$ is the dynamic representation of the previous time t_{i1}. $f(x)$ is a activation function, here we choose a *sigmoid* function $f(x) = \frac{1}{1+e^{-x}}$. Finally the model can output a users scores $\mathbf{o}_{u,t_i}$ towards all items at time t_i. The output $\mathbf{o}_{u,t_i}$ can be calculated through multiplication of item matrix $\mathbf{N}$ and a users dynamic representation $h^u_{t_{i-1}}$, which is formulated as follows:

$$\mathbf{o}_{u,t_i} = \mathbf{N}^T \mathbf{h}^u_{t_{i-1}}$$

. Thus, the current score is calculated by taking the previous dynamic representation of the user all the items that are closest to that representation.

B. Modified Model

In the modified model, we used the idea of attention from seq2seq encoder decoder architecture [7]. The intuition behind attention was to effectively select the most appropriate user's representation in the past in order to model the current hidden representation of the user and predict the next basket for the user. Over here our hypotheses was that the attention could be based on the notion of suggesting raw ingredients of a particular recipe that user might want to cook.

Fig. 5: Basic DREAM Model from paper [2]

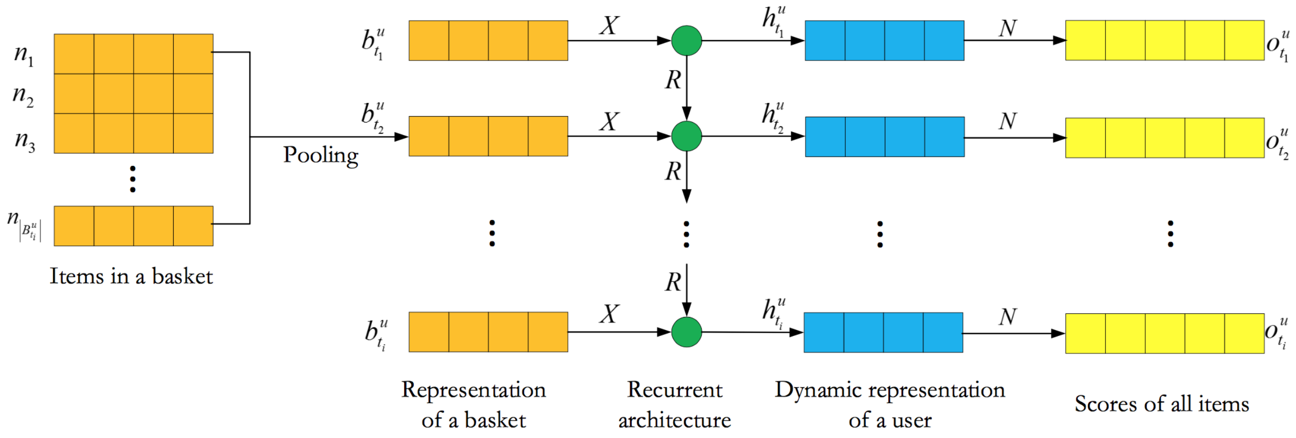
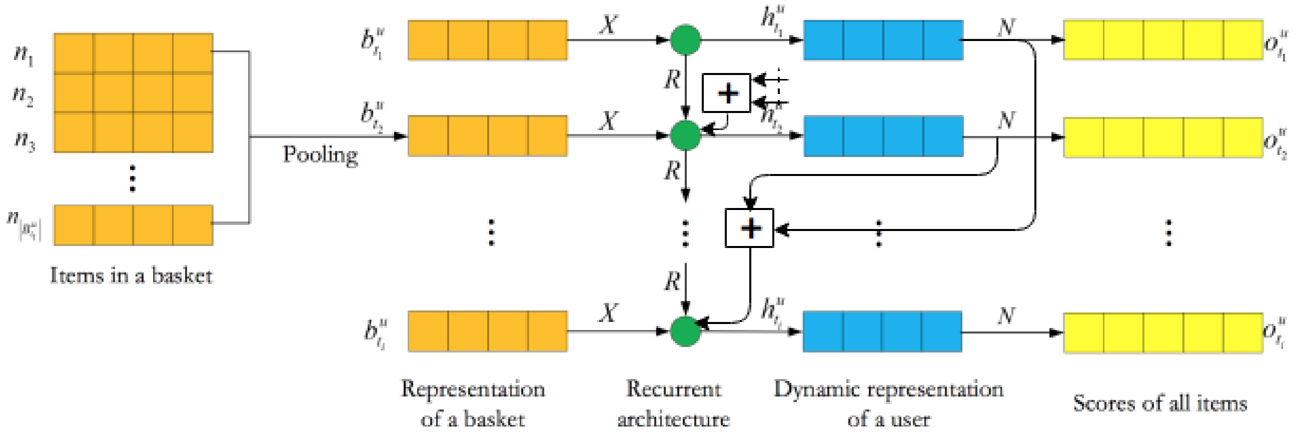


Fig. 6: Proposed Model: DREAM with Attention



In order to apply attention to the past user's representation, we select past l hidden representation of the user, assuming that l representation are sufficient to determine the current representation of the user as well as to predict the current basket. In a new model architecture, as can be seen from Figure 6 we compute the context vector c_i to be the weighted sum of the past l hidden representation of the user as:

$$c_{t_i} = \sum \alpha_{t_i,j} \mathbf{h}^u_j$$

where $\alpha_{t_i,j}$ represents the importance of the j-th hidden representation to compute the t_i-th hidden representation and o_{t_i}. We compute $\alpha_{t_i,j}$ as:

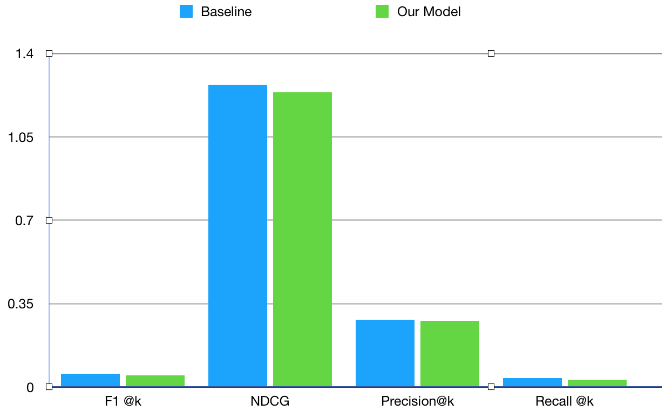
$$\alpha_{t_i,j} = \frac{exp(e_{t_i,j})}{\sum_k exp(e_{t_i,k})}$$

where $e_{t_i,k} = a(s_{t_{i-1}}, h^u_k)$, where $s_{t_{i-1}}$ represents the cell state of the LSTM, and a represents a feed-forward neural network with a single layer. Intuitively, $e_{t_i,k}$ is an alignment model which scores how well the inputs around position k and the output at position t_i match. The score is based on the LSTM hidden state $s_{t_{i1}}$ and the k-th annotation h^u_k of the user's u hidden representation.

Intuitively, this implements a mechanism of attention in the basket recommendation. The current $h^u_{t_i}$ decides parts of the previous hidden representation to pay attention to. With this new approach the information can be spread throughout the sequence of annotations, which can be selectively retrieved by the $h^u_{t_i}$ accordingly.

V. Evaluation

Fig. 7: Model Evaluation against various metrics



We use Bayesian Personalized Ranking (BPR) during the learning phase of the model. BPR is a state-of-the-art pairwise ranking framework for the implicit feedback data. The basic assumption is that a user prefers an item in basket at a specific time than a negative item sample. The negative items can be any other items apart from those in the basket. In this way, we need to maximize the following probability:

$$p(u,t,v>v) = \sigma(o_{u,t,v} - o_{u,t,v'})$$

where v' denotes a negative item sample, and $\sigma(x)$ is a non- linear function which is chosen as $sigma(x) = \frac{1}{1+e^{-x}}$. Adding up all the log likelihood and the regularization term, the objective function can be written as follows:

$$J = \sum \log(1+e^{-(o_{u,t,v}-o_{u,t,v'})}) + \frac{\lambda}{2}||\theta||^2,$$

where $\theta = \{\mathbf{N},\mathbf{R},\mathbf{X}\}$ denotes all the parameters to be learnt, λ is a parameter to control the power of regularization. Furthermore, the objective function can be optimized by Back Propagation Through Time (BPTT). BPTT is to iteratively repeat the calculation of derivations of J with respect to different parameters and obtain these gradients of all the parameters in the end. Then we update parameters utilizing Stochastic Gradient Descent (SGD) until converge.

Notice that the DREAM model utilize an iterative method in learning users' representation vectors. That is to say, for any new transactions, we can update users' representation vectors based current ones. Some state-of-the-art models, such as HRM, need to factorize a new built user-item matrix to get users' representation vectors. Therefore this iterative learning method may be more practical in real-world applications.

VI. Evaluation metrics and setup

A. Setup

Since, the original datasets used in the paper was unavailable, we used a similar dataset of Instacart grocery items. We implemented the baseline model in Pytorch and ran the DREAM model on Instacart dataset. The Instacart dataset was huge and had 3.4 million anonymized grocery orders from more than 206,000 users. We sampled 10% of the data and trained our model on a subset dataset which had 32,000 users, 528,000 transactions or orders and 44440 items. The original paper used item and user embedding of size 64. But since, we were only training on a sampled dataset, we downgraded the embedding size to 32. We used 80:20 split of train and test data. Our baseline model converged in 10 epochs, where every epoch took approximately 500s to train our model.

For the extended model, we added attention layer which performs a softmax operation over the past l user representations. We experimented over different values of l and found l=10 to work best for our use case. For training on the extended model, we kept the setting of all the other hyper parameters as the same .

B. Evaluation metrics

While testing, we generate a list of K items (K=5) for each user u. To perform the evaluation of the recommended next basket, we use 4 evaluation metrics, i.e. Precision@k, recall@k, F1-score@k and Normalized Discounted Cumulative Gain(NDCG). Precision@k measures the number of correct items predicted by the model divided by the number of items predicted. Recall@k measures the number of correct items predicted by the model divided by the basket size. F1-score@k calculates the harmonic mean of the precision@k and recall@k. NDCG is a cumulative measure of ranking quality, which is more sensitive to the relevance of higher ranked items. For all the metrics, the larger the value, the better the performance.

We use 20% of the unseen users to test our model. Yu et al initializes the items and users embeddings randomly. After the training, the model learns both these embeddings which are then multiplied to get the list of the items user is most likely to buy next.

TABLE I: Evaluation metrics

Model	F1@k	NDCG	Precision@k	Recall@k
Baseline	0.0548	1.2688	0.2822	0.0367
Our Model	0.0493	1.2377	0.2767	0.0303

VII. Results

The table below shows the comparison of the results obtained by the baseline model and our extension of adding attention.

The results show that attention was not able to improve on the baseline for the current hyperparameters and training time. This can be attributed to less/no finetuning of the hyper parameters due to time constraints. Since, in adding attention to our baseline model, we did not remove any of the previous connections, we only added another component to the network. Hence, if the system could produce better results by not using this additional component, the system would learn to by pass this added component. The reason our results show otherwise, can be because the network changed, hence the ideal hyper parameters and learning time for this new system have changed.

The results are also shown in a bar chart 7 to better visualize the difference in performance of the baseline and our extended model.

VIII. Conclusion

In this assignment, we solved the problem of predicting the next basket for a user using the history of user orders. The baseline model was based on a RNN model which learnt the internal representation of the items and users. To generate the ranked list of items for a user, we take the dot product of the user and item embeddings. We added 2 extensions to the baseline. One, we used a different dataset. Second, we added attention over the past latent representations of the users. The hidden representations of the users are captured at each time step t. The attention is based on an alignment score, i.e, how correlated is the current input to each of the previous baskets in a window. In the current experimental setting, attention degraded the performance of the baseline model. Though, we believe that further hyper parameter tuning would enhance the model's performance.

IX. Future Work

In the light of the experiments we performed, there are multiple things which can be explored in the future. First of all, due to resource constraints we used only a subset of the total dataset, it is worth experimenting with the output of the model on the complete dataset. Second, due to time constraints, we were not able to finetune the hyper parameters for our use case. Third, we only used the past user orders for predicting the next basket, but the dataset contains several other features which can be used in conjunction with the latent representations of the user and item to better predict the next basket of the user. Fourth, adding attention to give differential importance to past user representations did not work for our use case. But it is worth visualizing the results of the network predictions after adding attention to see how the model predictions actually change from the baseline.

References

[1] Instacart dataset
https://www.instacart.com/datasets/grocery-shopping-2017

[2] A Dynamic Recurrent Model for Next Basket Recommendation, Feng Yu, Qiang Liu, Shu Wu, Liang Wang, Tieniu Tan
http://delivery.acm.org/10.1145/2920000/2914683/p729-yu.pdf

[3] Matrix factorization techniques for recommender systems Y. Koren, R. Bell, and C. Volinsky
http://ceur-ws.org/Vol-1441/recsys2015_poster15.pdf

[4] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In WWW, pages 811820, 2010

[5] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, and X. Cheng. Learning hierarchical representation model for nextbasket recommendation. In SIGIR, pages 403412, 2015

[6] Shengxian Wan, Yanyan Lan, Pengfei Wang, Jiafeng Guo, Jun Xu, Xueqi Cheng Next Basket Recommendation with Neural Networks

[7] Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio Neural Machine Translation by Jointly Learning to Align and Translate