Collaborative Multi-modal deep learning for the personalized product retrieval in Facebook Marketplace

Lu Zheng
Facebook
1 Hacker Way
Menlo Park, California 94025
newluzheng@gmail.com

Zhao Tan
Facebook
1 Hacker Way
Menlo Park, California 94025
maximtzh1000@gmail.com

Kun Han
Facebook
1 Hacker Way
Menlo Park, California 94025
hankun@fb.com

Ren Mao
Facebook
1 Hacker Way
Menlo Park, California 94025
neroam@fb.com

ABSTRACT

Facebook Marketplace [1] is quickly gaining momentum among consumers as a favored customer-to-customer (C2C) product trading platform. The recommendation system behind it helps to significantly improve the user experience. Building the recommendation system for Facebook Marketplace is challenging for two reasons: 1) Scalability: the number of products in Facebook Marketplace is huge. Tens of thousands of products need to be scored and recommended within a couple hundred milliseconds for millions of users every day; 2) Cold start: the life span of the C2C products is very short and the user activities on the products are sparse. Thus it is difficult to accumulate enough product level signals for recommendation and we are facing a significant cold start issue. In this paper, we propose to address both the scalability and the cold-start issue by building a collaborative multi-modal deep learning based retrieval system where the compact embeddings for the users and the products are trained with the multi-modal content information. This system shows significant improvement over the benchmark in online and off-line experiments: In the online experiment, it increases the number of messages initiated by the buyer to the seller by +26.95%; in the off-line experiment, it improves the prediction accuracy by +9.58%.

CCS CONCEPTS

•Information systems → Retrieval models and ranking; •Computing methodologies → Machine learning;

KEYWORDS

Neural network, Facebook, content based retrieval, recommender system

ACM Reference format:
Lu Zheng, Zhao Tan, Kun Han, and Ren Mao. 2016. Collaborative Multi-modal deep learning for the personalized product retrieval in Facebook Marketplace. In *Proceedings of ACM Conference, Washington, DC, USA, July 2017 (Conference'17),* 6 pages.
DOI: 10.475/123_4

1 INTRODUCTION

Facebook rolled out Marketplace where people can buy, sell and discover products [1] (Figure 1). After the launch, Facebook Marketplace quickly obtain momentum among the users as a favored C2C product trading platform.

The Marketplace recommendation system is responsible for helping users to discover products that best fit their interests. Due to the large traffic, the Marketplace recommendation system needs to process a huge number of requests every day. A common way of handling the mass recommendation in the industry is dividing the recommendation system into two cascade systems – a retrieval system and a ranking system. The retrieval system is responsible for returning a short list from the candidate pool; The ranking system is responsible for giving the final recommendations out of the short list returned by the retrieval system. In the Marketplace recommendation system, we adopt this ranking-retrieval paradigm.

In this paper, we will focus on how we use multi-modal deep learning to build the content based retrieval system. The retrieval stage recommendation is challenging for the following reasons:

- *Real-time:* The retrieval system needs to serve millions of users to explore millions of the products in real time. The computation should be finished within a required latency (often on the order of O(100) milliseconds). Highly specialized and efficient serving system is needed for handling the Marketplace product retrieval.
- *Lack of the product taxonomy:* Most of the sellers on Facebook Marketplace are non-professional sellers. User research has shown that for these sellers, asking for too much product information will decrease their posting intent. Therefore, we only require the sellers to provide a coarse/broad category for the products when they post the products. This results in the lack of the accurate/fine

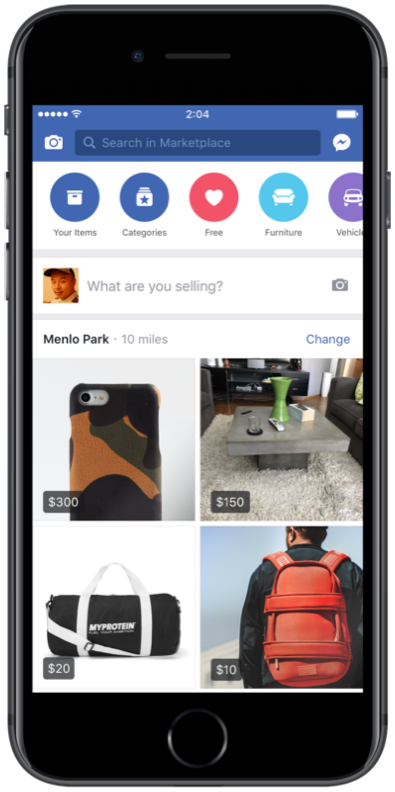


Figure 1: Recommendation for the Facebook Marketplace.

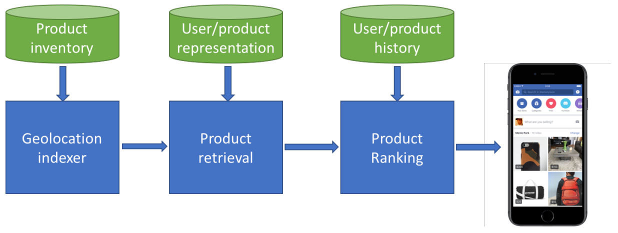


Figure 2: Two stage recommender system for Marketplace

grained product taxonomy for Facebook Marketplace and it is difficult to build a structured index for the products.

- *Cold start and sparse user feedback:* Due to the huge inventory size and the buyers' preference of the newly posted products over the old products, the life spans of the C2C products are very short. A typical C2C product is active only for a few days before they fade out of Marketplace. It is difficult to collect enough product level signal within this short life span.

However, there is a silver lining to the above challenges: in Marketplace, the content of the products, including the product images and the text descriptions, plays an important role in driving the buyer's purchase intent. With that, we are inspired to build a retrieval system that extensively explores the relationship between the product content and the user activity. The core of this retrieval system is two collaboratively trained user and product deep neural network (DNN) models. The models will digest the user profile and the product content to generate the user and the product representations. The product representations are used to index the products. At run time, the similarity between the user and the product representation is computed to retrieve the most relevant products for the user.

There are numerous works about recommendation systems. Existing recommendation systems can be roughly divided into three categories: collaborative filtering based approach[18], content based approach and the hybrid method approach. Collaborative filtering approach [16, 17] makes use of the user activities or feedback without using the item content. The content based approach make use of the user profile and the item description for recommendation [10]. The hybrid approach [2, 8, 11] is a combination of the content based approach and the collaborative filtering based approach.

Deep learning has successful applications in various areas such as computer vision (CV), image processing[15] and the natural language processing (NLP) [5]. Recently, there are also works applying deep learning in mass scale recommendation system [3, 6]. The recommendation is modeled as an extreme multi-class classification problem in [6] and the user embeddings are learned based on the user's implicit feedback.

While we also apply deep learning in the retrieval system, we focus on exploring the power of the deep learning model for the content understanding. As mentioned above, the reasons for this are (1) in Marketplace, the users' purchase intent is driven by the content of the products. (2) the user feedback for the products is sparse due to the short life-span of the products. (3) The content information is relatively static compared with the history information. It is therefore easier to design the service system.

This paper is organized as follows: a brief system overview is presented in Section 2. Section 3 describes the retrieval system in details. In Section 4, we provide the evaluation for the retrieval model.

2 SYSTEM OVERVIEW

Facebook Marketplace is designed to enable users to discover and purchase nearby products. An overview of the recommendation system behind Marketplace is shown in Figure 2. The system consists of a geo-location indexer, a retrieval system and a ranking system. The geo-location indexer indexes all Marketplace products based on the product geo-location so that we can easily look up the nearby products for the users. The retrieval system generates a list of the products from the local product pool. The ranking system then ranks the retrieved products to best match the context and the user interests.

The retrieval system sits between the geo-location indexer and the ranking system. In a Marketplace recommendation request, a user and his/her interests are viewed as a query. The Marketplace retrieval system is designed to match the products with the queries based on the content of the products.

3 THE RETRIEVAL SYSTEM DESIGN

Table 1: Notations

Notation	Description
$\mathcal{U}$	User set
$\mathcal{I}_u$	Product Item set available for the user u
$i, j \in \mathcal{I}$	product index
$u \in \mathcal{U}$	user index
Θ	Model parameter

Given a user u and a set of products that satisfy the this user's geo-location setting $\mathcal{I}_u$, the retrieval result $\mathcal{I}_u^*$ is set to be:

$$\mathcal{I}_u^* = Top\_N_{i \in \mathcal{I}_u}(r_{ui}), \tag{1}$$

where r_{ui} is the score between the user u and the product i, N is the total number of products retrieved.

Since we need the serving system to be efficient, r_{ui} is modeled as a dot product of a user "embedding" and a product "embedding", i.e., $r_{ui} = \phi(u) * \gamma(i)$ where ϕ and γ represent the mapping from the user and the product to the latent embedding space. At serving time, we can simply perform a nearest neighbor search to obtain the top N products for users.

Now the problem is reduced to modeling the ϕ and γ. In this paper, ϕ and γ will be collaboratively learned from the user and the product interaction. Given the run-time environment constraint, we want the models of ϕ and γ to be as computationally light-weight as possible. Therefore, we choose to learn ϕ and γ on relative static information to avoid the need for the frequent embedding update. For ϕ, the input includes the *user demographics* and the *long term click history*; for γ, the input includes multiple modalities such as *product images* and the *product text*. We will discuss the details of how we model ϕ and γ modeling in the following sections.

3.1 Collaborative deep learning with pairwise rank loss

Since we want to learn the relative user preference for one item over another, we adopt *pairwise collaborative filtering* to train the Marketplace retrieval models. Pairwise collaborative filtering with implicit user feedback is studied in [14]. Empirically, this approach is better than point-wise collaborative filtering [13].

The *pairwise preference over the items* is defined as follows:

$$r_{ui} > r_{uj}, i, j \in \mathcal{I}_u, \tag{2}$$

where the relationship $r_{ui} > r_{uj}$ means that the user u prefer the item $i \in \mathcal{I}_u$ to the item $j \in \mathcal{I}_u$. In the ranking context, we relax the *pairwise preference over items* to the *pairwise preference over the item sets within a margin*[14]:

$$r_{u\mathcal{P}_+} - r_{u\mathcal{P}_-} > \xi, \mathcal{P}_+ \subset \mathcal{I}_u, \mathcal{P}_- \subset \mathcal{I}_u, \tag{3}$$

where $r_{u\mathcal{P}_+}$ and $r_{u\mathcal{P}_-}$ are the user u's overall ranking scores on the items from the item-set $\mathcal{P}_+$ and $\mathcal{P}_-$, respectively. Here $\mathcal{P}_+$ and $\mathcal{P}_-$ depend on the user u and should be represented as $\mathcal{P}_+(u)$ and $\mathcal{P}_-(u)$. For simplicity we drop this dependency in the equation for the rest of this paper. ξ is the margin, which is set to 1 in this paper.

In our work, we will combine the deep learning model with pairwise collaborative filtering to learn ϕ and γ.

The overall model is shown in Figure 3. The model consists of two parts: a user model $\phi(u)$ and a product model $\gamma(i)$. They are jointly trained on the events logging of the products: We label an impression of the product to be positive or negative based on whether the impression led to any messages sent from the buyer to the seller. We use the pairwise rank loss to guide the training of the model. We call the model pairNN for short.

3.1.1 *pairwise loss function.*

Let Θ be the model parameter. The objective function for the training is:

$$\min_{\Theta} \sum_{u \in \mathcal{U}} \sum_{p_+ \in \mathcal{P}_+} \sum_{p_- \in \mathcal{P}_-} \max(0, r_{up_-} - r_{up_+} + \xi), \tag{4}$$

For each user u, we consider the product p as a positive example p_+ if this user initiate a message thread with the seller about this product and a negative example p_- otherwise. Therefore, a training example is a tuple of (u, p_+, p_-).

In practice, with the data from all the users, we collaboratively train the model by optimizing the following objective function:

$$\min_{\Theta} \sum_{(u, p_+, p_-) \in \mathcal{E}} \max(0, r_{up_-} - r_{up_+} + \xi), \tag{5}$$

where $\mathcal{E}$ presents all the training samples.

In order to compute the ranking scores r_{up_+} and r_{up_-}, we use deep neural networks (DNNs) to learn the embeddings for the user and the product separately, and then use the cosine similarity between the embeddings as the ranking score.

3.1.2 *User model.*

We design the user model to take the heterogeneous information including the user keywords and the user demographic features.

In the modeling, we use an embedding layer to map each keyword to a vector. The embedding is initialized with a set of pre-trained word2vec embeddings [12] and updated during the training. An average pooling layer is used to aggregate the embedding vectors of all keywords. The output of the averaged word embeddings is concatenated with the user demographic information in the form of the dense feature, and then sent into a multilayer perceptron (MLP) to learn the user embedding vector.

3.1.3 *Product model.*

The features of the product model include the text and the images of the products. The text feature is the word sequence of the product title and description. To model the word sequence, we first map each word in the word sequence to an embedding and then feed them to the convolutional neural networks (CNNs) [4, 9]. The word sequence is then converted into a fixed length text embedding. For image feature, we utilize a Facebook internal image classification model to encode the images of the products. The image encoder is a 50-layer ResNet [7] pre-trained on the image classification task and stays fixed in the training procedure. The output of the image encoder and the text embedding are concatenated and sent into a multilayer perceptron (MLP).

With the vector representations of both the user and the product, we compute the cosine similarity as the final score r_{ui}. At training time, we optimize the similarity score with a ranking loss defined in Eq. 5.

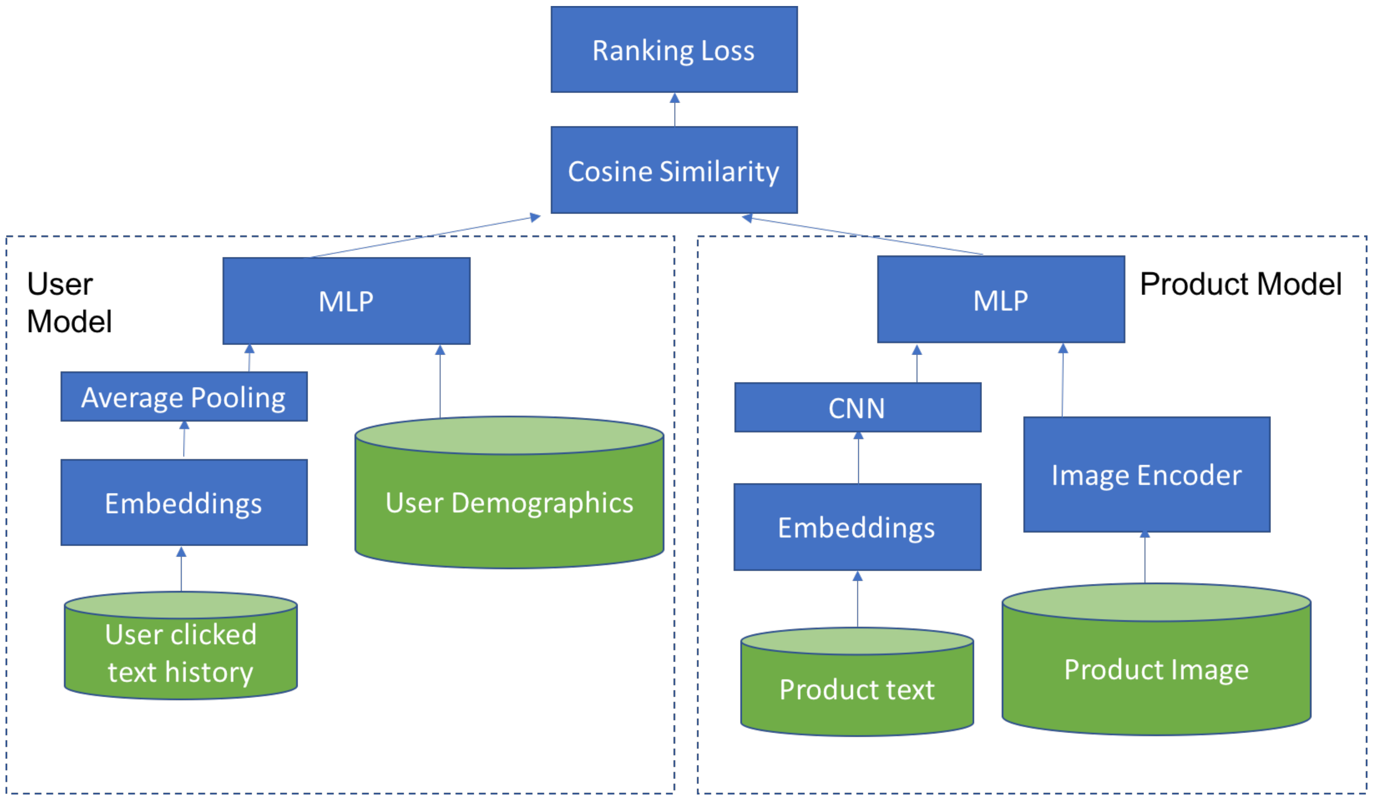


Figure 3: Deep Neural Network structure for User and Product

4 EXPERIMENTS AND RESULTS

4.1 Training Data

We generate the training data from the user message log through Facebook Marketplace. The data we collected is limited and anonymized. The content of the message is unaccessible to the researchers – we only log whether a message is initiated by a buyer to a seller to label whether the product is a positive or negative example for this buyer. A total of 5+M messages are sampled from a couple of weeks message log as the positive examples. An additional 70+M impressions are sampled as the negative examples. Since we are using the pair-wise loss function in the training, the required training sample is a tuple (u, p_+, p_-), where $u \in \mathcal{U}$, $p_+ \in \mathcal{P}_+$ and $p_- \in \mathcal{P}_-$. Thus we pair up one positive example and one negative example for a given user and form the pair-wise training data.

4.2 Experiment Setup

We evaluate the models with both the online and the offline experiment.

The online experiment evaluates the performance of the embeddings in the online retrieval system for Marketplace. In the retrieval task, for user u, we will first fetch the most recent M products from the set of I_u that satisfies the user's location and radius setting and then select the top N products based on the score r_{ui} out of the M products. The selected N products are then used as the input to the ranking system.

The results are evaluated by measuring the number of messages initiated by the buyers in the A/B test groups. The system randomly choose de-identified users on Facebook Marketplace platform and split them into the test and the control groups for A/B test. For different groups we are using different approaches to retrieve product candidates. The ranking stage is the same across different groups in the experiment though.

The offline experiment evaluates the model in a classification setting: the evaluation data is generated the same way as the training data where we pair up the positive p_+ and negative samples p_- for a given user u. For an evaluation sample (u, p_+, p_-), we call the classification correct if $r_{up_+} > r_{up_-}$ and wrong otherwise.

4.3 Metrics

In the online experiment, we measure the performance of the model based on the message initiation by the buyers.

- **Message initiation:** We count the number of initial messages sent from buyers to sellers through Facebook Marketplace.

In the offline experiment, the model is measured by the **accuracy** and the **average loss** defined as follows.

- **Accuracy:** Considering a binary classifier on the dataset consists of tuple (u, p_+, p_-), the accuracy of models is defined as, $Acc = \sum_{(u,p_+,p_-)\in\mathcal{T}} \delta(r_{up_+} - r_{up_-})/|\mathcal{T}|$, where $\delta(x) = 1$ if $x > 0$ and $\delta(x) = 0$ otherwise. Thus $\sum_{(u,p_+,p_-)\in\mathcal{T}} \delta(r_{up_+} -$

r_{up_-}) denotes the number of tuple samples in the test set that have higher score for positive product than negative one. With higher accuracy, the model in general can retrieve more positive products then negative ones.

- **Average loss** The average loss is defined as **average loss** = $\sum_{(u,p_+,p_-)\in\mathcal{T}} \max(0, r_{up_-} - r_{up_+} + \xi)/|\mathcal{T}|$. The average loss measures how well the model can distinguish the positive and the negative samples.

4.4 Baseline

We compare the pairNN based retrieval with two other retrieval approaches namely, time-based retrieval and word2vec-based retrieval.

- **Time-based Retrieval**: The time-based retrieval is the most nature way of retrieving the products. In this approach, the most recent M products are selected and fed to the ranking system.
- **Word2vec-based Retrieval**: In this approach, we fetch the products based on the cosine similarity between the user keywords and the product title keywords. The user keywords are again collected from the posts that the user clicked. We first train the word2vec keyword embeddings on a corpus of the product title and description. Given a list of user keywords $(k_1, ...k_m)$ and a list of product title keywords $(t_1, ...t_n)$, we map each of the user and product title keywords to the word2vec embedding $(v_{k_1}, ...v_{k_m})$ and $(v_{t_1}, ..., v_{t_n})$. The text similarity is computed as follows:

$$similarity(user, product) = \bar{v}_k * \bar{v}_t \tag{6}$$

where $\bar{v}_k = \sum_{i=1}^{m} v_{k_i}/m$ and $\bar{v}_t = \sum_{i=1}^{n} v_{t_i}/n$. We retrieve the top N products based on the similarity score.

4.5 Summary of the experiment result

In the offline experiment, We compare the **accuracy** and the **average loss** of the pairNN model and the word2vec-based model. The pairNN model is trained on the 10+ days of Facebook Marketplace message data and the world2vec model is trained on a Marketplace product description corpus.

In pairNN, the MLP blocks in the user model and the product model both have three layers: the input layer has 100 perceptrons; the hidden layer has 100 perceptrons and the output layer has 50 perceptrons. On the product side, the image encoder is a 50 layer pre-trained ResNet.

The results are summarized in Table 2. From the offline results, we can see that:

- Given the text only based approach, pairNN has better accuracy and average loss. This matches our expectation since the pairNN is trained supervisedly on the product event data while the word2vec embeddings are trained unsupervisedly.
- We compare the performance of pairNNs trained with different product content as the input: a) the text including the product title and description. b) the user uploaded image for this product c) the text and the image. From Table 2 we can see the multi-modal model generates the best accuracy.

Table 2: Offline result

	Accuracy	Average Loss
Word2vec-based	0.6312	0.8837

PairNN	Accuracy	Average Loss
Text only	0.6656	0.7598
Image only	0.6621	0.7650
Text + Image	**0.6917**	**0.7239**

The learning curves and distribution of prediction scores of the multi-modal model on the test dataset are shown in Fig. 4.

In the online A/B experiment, we compare the multi-modal pairNN based retrieval system, the word2vec-based retrieval system and the time based retrieval. The word2vec-based retrieval and time based retrieval are described in Section 4.4.

We measure the quality of the retrieval results by the number of messages initiated by the users. The online experiment ran for two weeks and Table 3 summarizes the experiment result. Compared with the time based retrieval, word2vec based retrieval increases the number of messages by 10.39% and the pairNN based retrieval increases the number of messages by 26.95%.

Table 3: Online Performance

Methods	#message
Time-based (baseline)	308k
Word2vec-based	340k ($\Uparrow$ 10.39%)
Proposed	391k ($\Uparrow$ 26.95%)

5 CONCLUSION AND FUTURE WORK

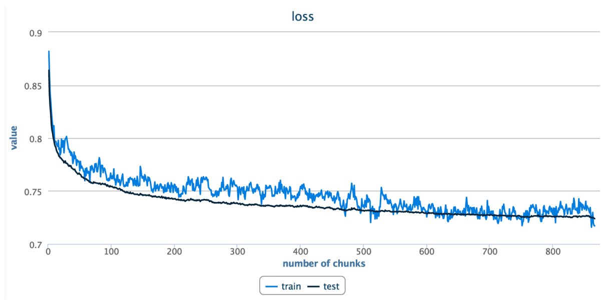
In this paper, we show the model architecture and the system design for Facebook Marketplace product retrieval. Given the real-time constraint and the large traffic volume, we need to extensively explore content information for the retrieval. In addition to that, the content based retrieval system can also help us resolve the cold start problem. The proposed pairNN model enable us to leverage the multi-modal information from products and shows significant improvement for the retrieval quality in both online and offline evaluation.

REFERENCES

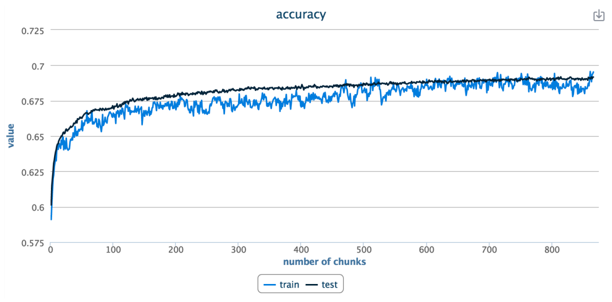
[1] 2017. Facebook launches Marketplace, a friendlier Craigslist. https://techcrunch.com/2016/10/03/facebook-marketplace-2/. (2017).

[2] Deepak Agarwal and Bee-Chung Chen. 2009. Regression-based Latent Factor Models. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*. ACM, New York, NY, USA, 19–28. https://doi.org/10.1145/1557019.1557029

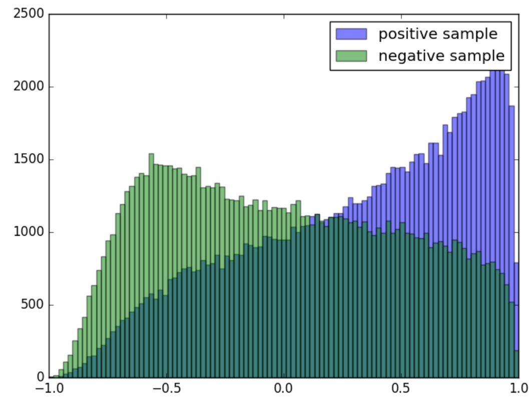
[3] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st*



(a) Loss curve



(b) Accuracy curve



(c) Distribution of prediction score in test set

Figure 4: Learning curve and Distribution of prediction score

Workshop on Deep Learning for Recommender Systems (DLRS 2016). ACM, New York, NY, USA, 7–10. https://doi.org/10.1145/2988450.2988454

[4] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

[5] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural Language Processing (almost) from Scratch. *CoRR* abs/1103.0398 (2011). http://arxiv.org/abs/1103.0398

[6] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16)*. ACM, New York, NY, USA, 191–198. https://doi.org/10.1145/2959100.2959190

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. *CoRR* abs/1512.03385 (2015). http://arxiv.org/abs/1512.03385

[8] Liang Hu, Jian Cao, Guandong Xu, Longbing Cao, Zhiping Gu, and Can Zhu. 2013. Personalized Recommendation via Cross-domain Triadic Factorization. In *Proceedings of the 22Nd International Conference on World Wide Web (WWW '13)*. ACM, New York, NY, USA, 595–606. https://doi.org/10.1145/2488388.2488441

[9] Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *In EMNLP*. Citeseer.

[10] Ken Lang. 1995. NewsWeeder: Learning to Filter Netnews. In *in Proceedings of the 12th International Machine Learning Conference (ML95*.

[11] Wu-Jun Li, Dit-Yan Yeung, and Zhihua Zhang. 2011. Generalized Latent Factor Models for Social Network Analysis.. In *IJCAI*, Toby Walsh (Ed.). IJCAI/AAAI, 1705–1710. http://dblp.uni-trier.de/db/conf/ijcai/ijcai2011.html#LiYZ11

[12] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[13] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N. Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-Class Collaborative Filtering. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM '08)*. IEEE Computer Society, Washington, DC, USA, 502–511. https://doi.org/10.1109/ICDM.2008.16

[14] Weike Pan and Li Chen. *CoFiSet: Collaborative Filtering via Learning Pairwise Preferences over Item-sets*. 180–188. https://doi.org/10.1137/1.9781611972832.20 arXiv:http://epubs.siam.org/doi/pdf/10.1137/1.9781611972832.20

[15] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *CoRR* abs/1506.01497 (2015). http://arxiv.org/abs/1506.01497

[16] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI '09)*. AUAI Press, Arlington, Virginia, United States, 452–461. http://dl.acm.org/citation.cfm?id=1795114.1795167

[17] Ruslan Salakhutdinov and Andriy Mnih. 2008. Probabilistic Matrix Factorization. In *Advances in Neural Information Processing Systems*, Vol. 20.

[18] Xiaoyuan Su and Taghi M. Khoshgoftaar. 2009. A Survey of Collaborative Filtering Techniques. *Adv. in Artif. Intell.* 2009, Article 4 (Jan. 2009), 1 pages. https://doi.org/10.1155/2009/421425