Meta-Prod2Vec - Product Embeddings Using Side-Information for Recommendation

Flavian Vasile
Criteo
Paris
f.vasile@criteo.com

Elena Smirnova
Criteo
Paris
e.smirnova@criteo.com

Alexis Conneau *
Facebook AI Research
Paris
aconneau@fb.com

ABSTRACT

We propose Meta-Prod2vec, a novel method to compute item similarities for recommendation that leverages existing item metadata. Such scenarios are frequently encountered in applications such as content recommendation, ad targeting and web search. Our method leverages past user interactions with items and their attributes to compute low-dimensional embeddings of items. Specifically, the item metadata is injected into the model as side information to regularize the item embeddings. We show that the new item representations lead to better performance on recommendation tasks on an open music dataset.

Keywords

Recommender systems; Embeddings; Neural Networks

1. INTRODUCTION

In the recent years, online commerce outpaced the growth of traditional commerce, with a rate of growth of 15% in 2015 and accounting for $1.5 trillion spend in 2014. The research work on recommender systems has consequently grown significantly during the last couples of years. As shown by key players in the online e-commerce space, such as Amazon, Netflix and Alibaba, the product recommendation functionality is now a key driver of demand, accounting in the case of Amazon for roughly 35% [2] of the overall sales.

As of now, the state-of-the-art recommendation methods include matrix factorization techniques of the item-item and user-item matrices that differ in the choice of weighting schemes of the matrix entries, the reconstruction loss functions and their choice with regards to the use of additional item content information. The real-world recommender systems have additional constraints that inform their architecture. Some of these major constraints include: scaling the recommender systems such that they can handle a huge amount of user interaction information, supporting real-time

---

*Alexis Conneau did this work while interning at Criteo.

changes in recommendation [5] and handling the cold-start problem [29].

In the last couple of years, a promising new class of neural probabilistic models that can generate user and product embeddings has emerged and has shown promising results. The new methods can scale to millions of items and show good improvements on the cold-start problem. In the context of product recommendation, they were successfully applied to ad recommendations in Yahoo! Mail [9], for Restaurant recommendations by OpenTable [1] and in the 1st prize winners in the 2015 RecSys Challenge [26].

In this paper, we present an extension of the *Prod2Vec* algorithm initially proposed in [9]. The *Prod2Vec* algorithm only uses the local product co-occurrence information established by the product sequences to create distributed representations of products, but does not leverage their metadata. The authors have proposed an extension [7] of the algorithm that takes into account the textual content information together with the sequential structure, but the approach is specific to textual metadata and the resulting architecture is hierarchical, therefore missing some of the side information terms by comparison with our method. In this work, we make the connection with the work on recommendation using side information and propose *Meta-Prod2Vec*, which is a general approach for adding categorical side-information to the *Prod2Vec* model in a simple and efficient way. The usage of additional item information as side information-only, e.g. available only at training time, is motivated by real-world constraints on the number of feature values a recommender system can keep in memory for real-time scoring. In this case, using the metadata only at training time keeps the memory footprint constant (assuming an existing recommendation system that uses item embeddings) while improving the online performance.

We show that our approach significantly improves recommendation performance on a subset of the 30Music listening and playlists dataset [30] with a low implementation and integration cost.

In Section 2 we cover previous related work and the relationship with our method. In Section 3 we present the *Meta-Prod2Vec* approach. In Section 4 we present the experimental setup and the results on the 30Music dataset. In Section 5 we summarize our findings and conclude with future directions of research.

2. RELATED WORK

Existing methods for recommender systems can roughly be categorized into collaborative filtering (CF) based meth-

ods, content-based (CB) methods and hybrid methods. CF-based methods [21] are based on user's interaction with items, such as clicks, and don't require domain knowledge. Content-based methods make use of the user or product content profiles. In practice, CF methods are more popular because they can discover interesting associations between products without requiring the heavy knowledge collection needed by the content-based methods. However, CF methods suffer from cold-start problem in which no or few interaction are available with niche or new items in the system. In recent years, more sophisticated methods, namely latent factor models, have been developed to address the data sparsity problem of CF methods which we will discuss in Section 2.1. To further help overcoming cold-start problem, recent works focused on developing hybrid methods by combining latent factor models with content information which we will cover in Section 2.2.

2.1 Latent factor models

Matrix factorization (MF) methods [22, 13] became popular after their success in the Netflix competition. These methods learn low-rank decompositions of a sparse user-item interaction matrix by minimizing the square loss over the reconstruction error. The dot product between the resulting user and item latent vectors is then used to perform recommendation.

Several modifications have been proposed to better align MF methods with the recommendation objective, for instance, Bayesian Personalized Ranking [24] and Logistic MF [11]. The former learns user and item latent vectors through pairwise ranking loss to emphasize the relevance-based ranking of items. The latter models the probability that a user would interact with an item by replacing the square loss in MF method with the logistic loss [11].

One of the first methods that learns user and item latent representations through neural network was proposed in [28]. The authors utilized Restricted Boltzmann Machines to explain user-item interaction and perform recommendations. Recently, shallow neural networks has been gaining attention thanks to the success of word embeddings in various NLP tasks, the focus being on Word2Vec model [18]. An application of Word2Vec to the recommendation task was proposed in [9], called Prod2Vec model. It generates product embeddings from sequences of purchases and performs recommendation based on most similar products in the embedding space. Our work is an extension of Prod2Vec and we will present its details in Section 3.1.

2.2 Latent factor models with content information

Many techniques have been used recently to create unified representations from latent factors and content information. One way to integrate user and item content information is to use it to estimate user and item latent factors through regression [3]. Another approach is to learn latent factors for both CF and content features, known as Factorization Machines [25].

Tensor factorization have been suggested as a generalization of MF for considering additional information [12]. In this approach, user-item-content matrix is factorized in a common latent space. The authors in [8] propose co-factorization approach where the latent user and item factors are shared between factorizations of user-item matrix

and user and item content matrices. Similar to [17], they also assigned weights to negative examples based on user-item content-based dissimilarity.

Graph-based models have also been used to create unified representations. In particular, in [33] user-item interactions and side information are modeled jointly through user and item latent factors. User factors are shared by the user-item interaction component and the side information component. Gunawardana et al. [10] leans the interaction weights between user actions and various features such as user and item metadata. The authors use a unified Boltzmann Machines to make a prediction.

3. PROPOSED APPROACH

3.1 Prod2Vec

In their *Prod2Vec* paper [9], Grbovic et al. proposed the use of the *Word2Vec* algorithm on sequences of product receipts coming from emails. More formally, given a set S of sequences of products $s = (p_1, p_2, ..., p_M)$, $s \in S$, the objective is to find a D-dimensional real-value representation $v_p \in R^D$ such that similar products are close in the resulting vector space.

The source algorithm - *Word2Vec* [18] is originally a highly scalable predictive model for learning word embeddings from text and belongs to the larger class of Neural Net Language Models [4]. Most of the work in this area is based on the Distributional Hypothesis [27], which states that words that appear in the same contexts have close if not identical meanings.

A similar hypothesis can be applied in larger contexts such as online shopping, music and media consumption and has been the basis of CF methods. In the CF setup, the users of the services are used as the distributed context in which products co-occur, leading to the classical item co-occurrence approaches in CF. A further similarity between co-count based recommendation methods and *Word2Vec* has been established by Omer et al. in [16]; the authors show that the objective of the embedding method is closely related to the decomposition of the matrix containing as entries the Shifted Positive PMI of the locally co-occurring items (words), where PMI is the Point-Wise Mutual Information:

$$PMI_{ij} = \log \left(\frac{X_{ij} \cdot |D|}{X_i X_j} \right)$$
$$SPMI_{ij} = PMI(i, j) - \log k$$

where X_i and X_j are items frequencies, X_{ij} is the number of times i and j co-occur, D is the size of the dataset and k is the ratio of negatives to positives.

Prod2Vec Objective.

In [23] the authors show that the *Word2Vec* objective (and similarly *Prod2Vec*'s) can be re-written as the optimization problem of minimizing the weighted cross entropy between the empirical and the modeled conditional distributions of context products given the target product (more precisely, this represents the Word2Vec-SkipGram model, which usually does better on large datasets). Furthermore, the prediction of the conditional distribution is modeled as a softmax

over the inner product between the target product and context product vectors:

$$
\begin{aligned}
L_{\mathrm{P2V}} &= L_{J|I}(\theta) \\
&= \sum_{ij}(-X_{ij}^{\mathrm{POS}}\log q_{j|i}(\theta) - (X_i - X_{ij}^{\mathrm{POS}})\log(1 - q_{j|i}(\theta)) \\
&= \sum_{ij} X_i(-p_{j|i}\log q_{j|i}(\theta) - p_{\neg j|i}\log q_{\neg j|i}(\theta)) \\
&= \sum_{i} X_i H(p_{\cdot|i}, q_{\cdot|i}(\theta)).
\end{aligned}
$$

Here, $H(p_{\cdot|i}, q_{\cdot|i}(\theta))$ is the cross-entropy of the empirical probability $p_{\cdot|i}$ of seeing any product in the output space J conditioned on the input product $i \in I$ and the predicted conditional probability $q_{\cdot|i}$:

$$
q_{j|i}(\theta) = \frac{exp(w_i^T w_j)}{exp(w_i^T w_j) + \sum_{j' \in (V_{J-j})} exp(w_i^T w_j')}
$$

where X_i is the input frequency of product i and X_{ij}^{POS} is the number of times the pair of products (i, j) has been observed in the training data.

The resulting architecture for *Prod2Vec* is shown in Figure 1, where the input space of all products situated in center window is trained to predict the values of the surrounding products by using a Neural Network with a single hidden layer and a softmax output layer.

However, the product embeddings generated by *Prod2Vec* only take into account the information of the user purchase sequence, that is the local co-occurrence information. Though richer than the global co-occurrence frequency used in Collaborative Filtering, it does not take into account other types of item information that are available (the items' metadata).

For example, assuming that the inputs are sequences of categorized products, the standard *Prod2Vec* embedding does not model the following interactions:

- given the current visit of the product p with category c, it is more likely that the next visited product p' will belong to the same category c

- given the current category c, it is more likely that the next category is c or one of the related categories c' (e.g. after a swimwear sale, it is likely to observe a sunscreen sale, which belongs to an entirely different product category)

- given the current product p, the next category is more likely to be c or a related category c'

- given the current category c, the more likely current products visited are p or p'

As mentioned in the introduction, the same authors extended the *Prod2Vec* algorithm in [7] such that to take into account both product sequence and product text in the same time. If applying the extended method to non-textual metadata, the algorithm models, additionally to the product sequence information, the dependency between the product metadata and the product id, but does not link the sequence of metadata and the sequence of product ids together.
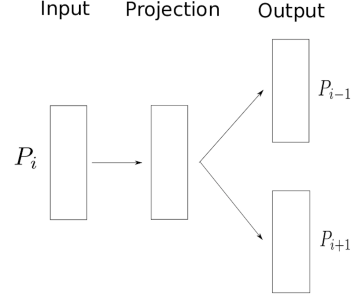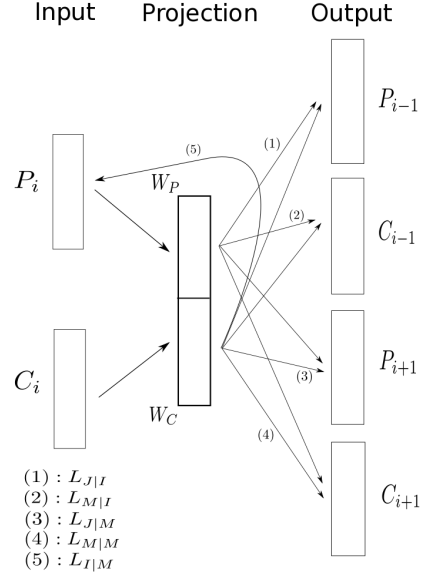

Figure 1: Prod2Vec Neural Net Architecture.

(1) : $L_{J|I}$
(2) : $L_{M|I}$
(3) : $L_{J|M}$
(4) : $L_{M|M}$
(5) : $L_{I|M}$

Figure 2: Meta-Prod2Vec Neural Net Architecture.

3.2 Meta-Prod2Vec

As shown in the Related Work section, there has been extensive work on using side information for recommendation, especially in the context of hybrid methods that combine CF methods and Content-based (CB) methods. In the case of embeddings, the closest work is the *Doc2Vec* [15] model where the words and the paragraph are trained jointly, but only the paragraph embedding is used for the final task.

We propose a similar architecture that incorporates the side information in both the input and output space of the Neural Net and parametrizes separately each one of the interactions between the items to be embedded and the metadata, as shown in Figure 2.

Meta-Prod2Vec Objective.

The *Meta-Prod2Vec* loss extends the *Prod2Vec* loss by taking into account four additional interaction terms involving the items' metadata:

$$
L_{MP2V} = L_{J|I} + \lambda \times (L_{M|I} + L_{J|M} + L_{M|M} + L_{I|M})
$$

	J	M	I
I	L_{P2V}	$\lambda_{mi}L_{M\|I}$	I_d
M	$\lambda_{jm}L_{J\|M}$	$\lambda_{mm}L_{M\|M}$	$\lambda_{im}L_{I\|M}$

Figure 3: MetaProd2vec as matrix factorization of the items and metadata extended matrix.

where: M is the metadata space (for example, artist ids in the case of the 30Music dataset), λ is the regularization parameter. We list the new interaction terms below:

$L_{I|M}$ is the weighted cross-entropy between the observed conditional probability of input product ids given their metadata and the predicted conditional probability. This side-information is slightly different from the next three types because it models the item as a function of its own metadata (same index in the sequence). This is because, in most cases, the item's metadata is more general than the id and can partially explain the observation of the specific id.

$L_{J|M}$ is the weighted cross-entropy between the observed conditional probability of surrounding product ids given the input products' metadata and the predicted conditional probability. An architecture where the normal *Word2Vec* loss is augmented with only this interaction term is very close to the *Doc2Vec* model proposed in [19] where we replace the document id information with a more general type of item metadata.

$L_{M|I}$ is the weighted cross-entropy between the observed conditional probability of surrounding products' metadata values given input products and the predicted conditional probability.

$L_{M|M}$ is the weighted cross-entropy between the observed conditional probability of surrounding products' metadata values given input products metadata and the predicted conditional probability. This models the sequence of observed metadata and in itself represents the *Word2Vec*-like embedding of the metadata.

To summarize, $L_{J|I}$ and $L_{M|M}$ encode the loss terms coming from modeling the likelihood of the sequences of items and metadata separately, $L_{I|M}$ represents the conditional likelihood of the item id given its metadata and $L_{J|M}$ and $L_{M|I}$ represent the cross-item interaction terms between the item ids and the metadata. In Figure 3 we show the relationship between the item matrix factorized by *Prod2Vec* and the one factorized by *Meta-Prod2Vec*.

The more general equation for *Meta-Prod2Vec* introduces a separate λ for each of the four types of side-information: λ_{mi}, λ_{jm}, λ_{mm} and λ_{im}.

In Section 4 we will analyze the relative importance of each type of side-information. Also, in the case when multiple sources of metadata are used, each source will have its own term in the global loss and its own regularization parameter.

In terms of the softmax normalization factor, we have the option of either separate the output spaces of the items and of their metadata or not. Similarly with the simplifying assumption used in *Word2Vec*, that allows each pair of co-occurring products to be predicted and fitted independently (therefore adding an implicit mutual exclusivity constraint on the output products given an input product), we embed the products and their metadata in the same space, therefore allowing them to share the normalization constraint.

One of the main attractions of the *Word2Vec* algorithm is its scalability, which comes from approximating the original softmax loss over the space of all possible words with the Negative Sampling loss [19, 20], that fits the model only on the positive co-occurrences together with a small sample of the negative examples to maximize a modified likelihood function $L_{SG-NS}(\theta)$:

$$L_{J|I}(\theta) = \sum_{ij}(-X_{ij}^{POS}\log q_{j|i}(\theta) - (X_{ij}^{NEG}\log(1 - q_{j|i}(\theta)))$$

$$\approx L_{SG-NS}(\theta)$$

and:

$$L_{SG-NS}(\theta) = \sum_{ij} -X_{ij}^{POS}(\log\sigma(w_i^T w_j)$$

$$-k\mathbb{E}_{c_N \sim P_D}\log\sigma(-w_i^T w_N))$$

where P_D probability distribution used to sample negative context examples and k is a hyper parameter specifying the number of negative examples per positive example. The side-information loss terms $L_{I|M}$, $L_{J|M}$, $L_{M|I}$, $L_{M|M}$ are computed according to the same formula, where the i,j indexes range over the respective input/output spaces.

In the case of *Meta-Prod2Vec*, the impact of the decision to co-embed products and their metadata on the $L_{SG-NS}(\theta)$ loss is that the set of potential negative examples for any positive pair ranges over the union of items' and metadata values.

Impact on final recommendation system. Feasibility and engineering concerns.

Because of the shared embedding space, the training algorithm used for *Prod2Vec* remains unchanged. The only difference is that, in the new version of the generation step of training pairs, the original pairs of items are supplemented with additional pairs that involve metadata. In terms of the online recommendation system, assuming we are augmenting a solution that already involves item embeddings, the online system does not incur any changes (since the only time we make use of the metadata is during training) and there is zero additional impact on the online memory footprint.

4. EXPERIMENTS

The experimental section is organized as follows. First, we describe the evaluation setup, namely, the evaluation task,

success metrics and the baselines. Then, we report results of experiments on the 30Music open dataset.

4.1 Setup

We evaluate the recommendation methods on the next event prediction task. We consider time ordered sequences of user interactions with the items. We split each sequence into the consequent training, validation and test sets. We fit the embedding *Prod2Vec* and *Meta-Prod2Vec* models on the first (n-2) elements of each user sequence and use the performance on the (n-1)-th element to bench the hyperparameters and we report our final by training on the first (n-1) items of each sequence and predicting the nth item. We use the last item in the training sequence as the *query item* and we recommend the most similar products using one of the methods described below.

As mentioned in Section 1, due to the technical constraint of keeping a constant memory footprint, we are interested in the usefulness of item metadata only at training time. Therefore we do not compare against methods where the metadata is used directly at prediction time, such as the supervised content-based embedding models proposed in [14] where both the user and item are represented as linear combinations of the item content embeddings and [32], where the products are represented by the associated image content embeddings.

We use the following evaluation metrics averaged over all users:

- Hit ratio at K (HR@K) that is equal to $1/K$ if the test product appears in the top K list of recommended products.

- Normalized Discounted Cumulative Gain (NDCG@K) favors higher ranks of the test product in the list of recommended products.

Using the aforementioned metrics, we compare the following methods:

- *BestOf*: this method retrieves the top products sorted by their popularity. This simulates the frequently encountered recommendation solution based strictly on popularity.

- *CoCounts*: the standard CF method which uses the cosine similarity of vectors of co-occurrences with other items. This method is performing particularly well in cases where the catalog of possible items is small and does not change a lot over time, therefore avoiding item cold-start problems.

- *Standalone Prod2Vec*: the product recommendation solution based on cosine similarities of the vectorial representation of products obtained by using *Word2Vec* on product sequences. Similarly with other embedding and matrix factorization solutions, the goal of the method is to address the cold-start problem.

- *Standalone Meta-Prod2Vec*: our proposed method, which enhances *Prod2Vec* with item side information and uses the resulting product embeddings to compute cosine similarities. As in *Prod2Vec*, the goal of the method is to further address cold-start problems.

Method	Pair freq=0	Pair freq<3
BestOf	0.0002	0.0002
CoCounts	0.0000	0.0197
Prod2Vec	0.0003	0.0078
Meta-Prod2Vec	**0.0013**	0.0198
Mix(Prod2Vec,CoCounts)	0.0002	0.0200
Mix(Meta-Prod2Vec,CoCounts)	0.0007	**0.0291**

Table 2: Recommendation accuracy (HR@20) in cold-start regime as a function of training frequency of the pair (*query item, next item*).

Side Information	% lift	
	HR@20	NDCG@20
Meta-Prod2Vec with only I|M	27%	32%
Meta-Prod2Vec with only M|I	50%	52%
Meta-Prod2Vec with only J|M	55%	60%
Meta-Prod2Vec without M|M	61%	65%

Table 3: Proportion of the Meta-Prod2Vec lift over BestOf due to each type of side information.

- *Mix(Prod2Vec,CoCounts)*: an ensemble method which returns the top items using a linear combination between the *Prod2Vec* and the *CoCount*-based item pair similarities. The motivation of the ensemble methods that combine embeddings and CF is to leverage the benefits of the two in cold-start and non cold-start regimes.

$$Mix(Prod2Vec, CoCounts) =$$
$$\alpha * Prod2Vec + (1-\alpha) * CoCounts \qquad (1)$$

- *Mix(Meta-Prod2Vec,CoCounts)*: an ensemble method which returns the top items using a linear combination between the *Meta-Prod2Vec* and the *CoCount*-based item pair similarity.

$$Mix(MetaProd2Vec, CoCounts) =$$
$$\alpha * MetaProd2Vec + (1-\alpha) * CoCounts \qquad (2)$$

4.2 Dataset

We perform our evaluation on the publicly available 30Music dataset [31] that represents a collection of listening and playlists data retrieved from Internet radio stations through Last.fm API. On this dataset, we evaluate the recommendation methods on the task of next song prediction. For the *Meta-Prod2Vec* algorithm we make use of track metadata, namely the artist information. We run our experiments on a sample of 100k user sessions of the dataset with the resulting vocabulary size of 433k songs and 67k artists.

4.3 Results

We keep the embedding dimension fixed to 50, window size to 3, the side information regularization parameter λ to 1. We bench the blending factor α in Equations 1,2 of the embedding-based similarity score with the *CoCounts*-based cosine similarity score and find that the best value $\alpha^* = 0.15$. In addition, we vary the number of training epochs and we find the best parameter value to be 30 for *Prod2Vec* and 10 for *Meta-Prod2Vec*. As shown in Table 1, *Meta-Prod2Vec* has better performance both standalone and

Method	HR@10	NDCG@10	HR@20	NDCG@20
BestOf	0.0003 (0.0002;0.0003)	0.001 (0.001;0.001)	0.0003 (0.0002;0.0003)	0.002 (0.002;0.002)
CoCounts	0.0248 (0.0245;0.0251)	0.122 (0.121;0.123)	0.0160 (0.0158;0.0161)	0.141 (0.139;0.142)
Prod2Vec	0.0170 (0.0168;0.0171)	0.105 (0.103;0.106)	0.0101 (0.0100;0.0102)	0.113 (0.112;0.115)
Meta-Prod2Vec	0.0191 (0.0189;0.0194)	0.110 (0.108;0.113)	0.0124 (0.0123;0.0126)	0.125 (0.123;0.126)
Mix(Prod2Vec,CoCounts)	0.0273 (0.027;0.0276)	0.140 (0.139;0.141)	0.0158 (0.0157;0.0160)	0.152 (0.151;0.153)
Mix(Meta-Prod2Vec,CoCounts)	**0.0292** (0.0288;0.0297)	**0.144** (0.142;0.145)	**0.0180** (0.0178;0.0182)	**0.161** (0.160;0.162)

Table 1: Comparison of recommendation performance of Meta-Prod2Vec and competing models in terms of HitRate and NDCG.
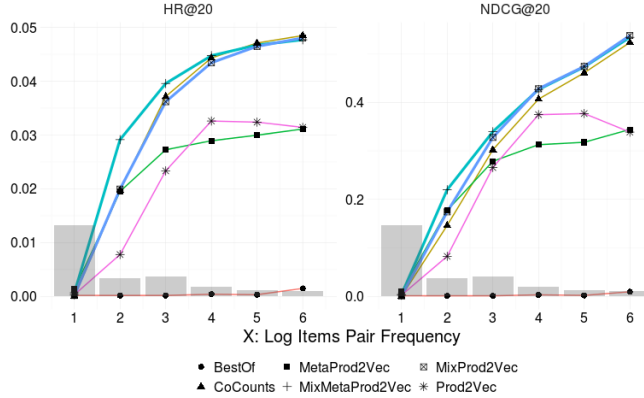

Figure 4: Cold-start improvements on the query and next item pairs.

in the ensemble model with respect to the *Prod2Vec* model (results computed at 90% confidence levels).

4.3.1 Improvements on Cold-Start

Most of the gains in performance are coming from the cold-start traffic, where, as shown in Figure 4 and Table 2, we can see that *Standalone Meta-Prod2Vec* outperforms by a large margin all other methods when the true pair of (query item, next item) have zero co-occurrences in the training set and that the *Mix(Meta-Prod2Vec,CoCounts)* has the best performance in the cases where the true pair has low observed counts. Interestingly, we observe that the difference of performance between the ensemble models is bigger than between the standalone *Meta-Prod2Vec* and *Prod2Vec* models. We explain this by the fact *Standalone Meta-Prod2Vec* outperforms *Standalone Prod2Vec* on the cold-start traffic therefore helping the overall performance of the *CoCounts*, that by itself performs really well on head traffic. Indeed, *CoCounts* memorizes frequent pairs of (query, target) product, while *Standalone Meta-Prod2Vec* helps to generalize on unseen ones. These results are mirrored by similar findings covered in [6] and motivate the newly introduced approach of Wide and Deep learning.

4.3.2 Relative Importance of each type of Side Information

We posited the relevance of each type of side information, but we want to confirm experimentally that each of the four types of side-information brings additional information. We proceed by introducing each of the types of the side infor-

mation separately and to compare its performance with the original *Prod2Vec* baseline.

The only exception for which we test the value by leaving it out of the full *Meta-Prod2Vec* model is the *mm* type of side-information; in this case, the input metadata explains the output metadata and this constraint is not directly valuable in regularizing the product embeddings and needs to be introduced together with the other types of pairs that connect the products to the metadata. Therefore its contribution to the model can be computed as (1 - degraded model performance) and is therefore 39% on HR and 35% on NDCG.

In Table 3, we compute the proportion of lift over the *BestOf* baseline obtained by using each type of side information separately and we observe that each one of them account for at most 50% of the performance of the full *Meta-Prod2Vec*, therefore confirming that the additional terms in our proposed model are relevant.

5. CONCLUSIONS

In this paper, we introduced *Meta-Prod2Vec*, a new item embedding method that enhances the existing *Prod2Vec* method with item metadata at training time. This work makes a novel connection between the recent embedding-based methods and consecrated Matrix Factorization methods by introducing learning with side information in the context of embeddings. We analyzed separately the relative value of each type of side information and proved that each one of the four types is informative. Finally, we have shown that *Meta-Prod2Vec* constantly outperforms *Prod2Vec* on recommendation tasks both globally and in the cold-start regime, and that, when combined with a standard Collaborative Filtering approach, outperforms all other tested methods. These results, together with the reduced implementation cost and the fact that our method does not affect the online recommendation architecture, makes this solution attractive in cases where item embeddings are already in use. Future work will extend on ways of using item metadata as side information and support of non-categorical information such as images and continuous variables.

6. REFERENCES

[1] Recsys 2015: Making meaningful restaurant recommendations at opentable. http://tinyurl.com/zs9at2t. Accessed: 2016-04-08.

[2] Venture beat article. http://venturebeat.com/2006/12/10/ aggregate-knowledge-raises-5m-from-kleiner-on-a-roll/. Accessed: 2016-04-08.

[3] D. Agarwal and B.-C. Chen. Regression-based latent factor models. In *Proceedings of the 15th ACM*

SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '09, pages 19–28, New York, NY, USA, 2009. ACM.

[4] Y. Bengio, H. Schwenk, J.-S. Senécal, F. Morin, and J.-L. Gauvain. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer, 2006.

[5] B. Chandramouli, J. J. Levandoski, A. Eldawy, and M. F. Mokbel. Streamrec: a real-time recommender system. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 1243–1246. ACM, 2011.

[6] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, et al. Wide & deep learning for recommender systems. *arXiv preprint arXiv:1606.07792*, 2016.

[7] N. Djuric, H. Wu, V. Radosavljevic, M. Grbovic, and N. Bhamidipati. Hierarchical neural language models for joint representation of streaming documents and their content. In *Proceedings of the 24th International Conference on World Wide Web*, pages 248–255. International World Wide Web Conferences Steering Committee, 2015.

[8] Y. Fang and L. Si. Matrix co-factorization for recommendation with rich side information and implicit feedback. In *Proceedings of the 2Nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, HetRec '11, pages 65–69, New York, NY, USA, 2011. ACM.

[9] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, and D. Sharp. E-commerce in your inbox: Product recommendations at scale. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1809–1818, New York, NY, USA, 2015. ACM.

[10] A. Gunawardana and C. Meek. A unified approach to building hybrid recommender systems. In *Proceedings of the Third ACM Conference on Recommender Systems*, RecSys '09, pages 117–124, New York, NY, USA, 2009. ACM.

[11] C. C. Johnson. Logistic matrix factorization for implicit feedback data. *Distributed Machine Learning and Matrix Computations*, 2014.

[12] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver. Multiverse recommendation: N-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, pages 79–86, New York, NY, USA, 2010. ACM.

[13] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, Aug. 2009.

[14] M. Kula. Metadata embeddings for user and item cold-start recommendations. In *Proceedings of the 2nd Workshop on New Trends on Content-Based Recommender Systems co-located with 9th ACM Conference on Recommender Systems (RecSys 2015), Vienna, Austria, September 16-20, 2015.*, pages 14–21, 2015.

[15] Q. V. Le and T. Mikolov. Distributed representations of sentences and documents. *arXiv preprint arXiv:1405.4053*, 2014.

[16] O. Levy and Y. Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems*, pages 2177–2185, 2014.

[17] Y. Li, J. Hu, C. Zhai, and Y. Chen. Improving one-class collaborative filtering by incorporating rich user information. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 959–968, New York, NY, USA, 2010. ACM.

[18] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[19] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[20] A. Mnih and K. Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2265–2273. Curran Associates, Inc., 2013.

[21] X. Ning and G. Karypis. SLIM: sparse linear methods for top-n recommender systems. In *11th IEEE International Conference on Data Mining, ICDM 2011, Vancouver, BC, Canada, December 11-14, 2011*, pages 497–506, 2011.

[22] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 502–511, Washington, DC, USA, 2008. IEEE Computer Society.

[23] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[24] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, pages 452–461, Arlington, Virginia, United States, 2009. AUAI Press.

[25] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidt-Thieme. Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 635–644, New York, NY, USA, 2011. ACM.

[26] P. Romov and E. Sokolov. Recsys challenge 2015: ensemble learning with categorical features. In *Proceedings of the 2015 International ACM Recommender Systems Challenge*, page 1. ACM, 2015.

[27] M. Sahlgren. The distributional hypothesis. *Italian Journal of Linguistics*, 20(1):33–54, 2008.

[28] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on*

Machine Learning, ICML '07, pages 791–798, New York, NY, USA, 2007. ACM.

[29] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 253–260. ACM, 2002.

[30] R. Turrin, M. Quadrana, A. Condorelli, R. Pagano, and P. Cremonesi. 30music listening and playlists dataset.

[31] R. Turrin, M. Quadrana, A. Condorelli, R. Pagano, and P. Cremonesi. 30music listening and playlists dataset. In *Poster Proceedings of the 9th ACM Conference on Recommender Systems*, Sept. 2015.

[32] A. Veit, B. Kovacs, S. Bell, J. McAuley, K. Bala, and S. Belongie. Learning visual clothing style with heterogeneous dyadic co-occurrences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4642–4650, 2015.

[33] S.-H. Yang, B. Long, A. Smola, N. Sadagopan, Z. Zheng, and H. Zha. Like like alike: Joint friendship and interest propagation in social networks. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 537–546, New York, NY, USA, 2011. ACM.