Word2vec applied to Recommendation: Hyperparameters Matter

Hugo Caselles-Dupré[1,2]*
[1] Flowers Laboratory (ENSTA ParisTech & INRIA)
[2] Softbank Robotics Europe
Paris, France
caselles@ensta.fr

Florian Lesaint
Deezer SA
Paris, France
flesaint@deezer.com

Jimena Royo-Letelier
Deezer SA
Paris, France
jroyo@deezer.com

ABSTRACT

Skip-gram with negative sampling, a popular variant of Word2vec originally designed and tuned to create word embeddings for Natural Language Processing, has been used to create item embeddings with successful applications in recommendation. While these fields do not share the same type of data, neither evaluate on the same tasks, recommendation applications tend to use the same already tuned hyperparameters values, even if optimal hyperparameters values are often known to be data and task dependent. We thus investigate the marginal importance of each hyperparameter in a recommendation setting through large hyperparameter grid searches on various datasets. Results reveal that optimizing neglected hyperparameters, namely negative sampling distribution, number of epochs, subsampling parameter and window-size, significantly improves performance on a recommendation task, and can increase it by an order of magnitude. Importantly, we find that optimal hyperparameters configurations for Natural Language Processing tasks and Recommendation tasks are noticeably different.

KEYWORDS

Recommender System Evaluation; Embeddings; Neural Networks

ACM Reference Format:
Hugo Caselles-Dupré[1,2], Florian Lesaint, and Jimena Royo-Letelier. 2018. Word2vec applied to Recommendation: Hyperparameters Matter. In *Twelfth ACM Conference on Recommender Systems (RecSys '18), October 2–7, 2018, Vancouver, BC, Canada.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3240323.3240377

1 INTRODUCTION

Word2vec (W2V) methods [16, 17] come from the Natural Language Processing (NLP) community. They were designed to produce low-dimensional distributional word representations. They were successfully applied to recommendation [8] to generate user and product embeddings as they can scale to millions of items.

Word corpora and sequences of items are two radically different type of data. Text data has a particular linguistic structure [5], constrained by grammatical and conjugating rules. Sequences of items, such as listening sessions or e-commerce purchase histories, have a different structure induced by the user's behaviour and the items' nature [9]. Moreover, linguistic and recommendation tasks are different. Intuitively, having accurate embeddings for popular items is crucial to perform on recommendation related tasks [24], where top items often represent most of the content users interacted with. On the contrary, for linguistic tasks, frequent words, mostly linking words, are not relevant [17] and so are their embeddings. Since hyperparameters choices are generally known to be data and task dependent [10], we expect optimal hyperparameter configuration to be different for NLP and recommendation.

W2V methods depend on several hyperparameters, some of which are already tuned to some extent by the algorithms' designers in order to perform well for NLP tasks such as word similarity and analogy detection [15], such that most renowned implementations [21?] set these values as default. In previous work that used W2V for recommendation [1, 8, 18–20, 26], the values of these hyperparameters are rarely discussed.

Thus, we study the marginal importance of each hyperparameter of Skip-gram with negative sampling (SGNS) in a recommendation setting, using Next Event Prediction (NEP) as an offline proxy for a recommendation task. We perform large hyperparameter grid searches on four different types of recommendation datasets (two of music, one of e-commerce and one of click-stream). This allows us to identify four hyperparameters, namely negative sampling distribution, number of epochs, subsampling parameter and window-size, which can significantly improve performance on the NEP task. This confirms that optimal values for these hyperparameters are data and task dependent, and that best configurations for recommendation tasks are radically different than those for NLP tasks, especially regarding the negative sampling distribution.

We first describe W2V methods and associated hyperparameters in Section 2. Then, we present the experiments in Section 3 and results in Section 4 before concluding in Section 5.

2 WORD2VEC

2.1 Methods

W2V [16, 17] is a group of word embedding algorithms that provides state-of-the-art results on various linguistic tasks [15]. They are based on the Distributional Hypothesis [23], which states that words that appear in the same contexts tend to purport similar meanings. The most common method, SGNS, used in the remaining of the paper, seeks to represent each word w as d-dimensional vector $\vec{w}$, such that words that are similar to each other have similar

*Work done while interning at Deezer

RecSys '18, October 2–7, 2018, Vancouver, BC, Canada

ACM ISBN 978-1-4503-5901-6/18/10…$15.00
https://doi.org/10.1145/3240323.3240377

vector representations. It does so by maximizing a function of products $\vec{w} \cdot \vec{c}$ where c appears in the context of w (a window around w of maximum size L), and minimizing the same function for negative examples (w, c_N) where c_N does not necessarily appear in the context of w. The loss function is

$$\ell_{\text{sgns}} = -\log(\sigma(\vec{w} \cdot \vec{c})) - \sum_{i=1}^{k} \log(\sigma(-\vec{w} \cdot \vec{c}_{N,i})), \quad (1)$$

where σ is the sigmoid function. For each observation of (w, c), SGNS forms k negative examples $(w, c_{N,i})_{i \in \{1,..,k\}}$ by sampling (hence the term "negative sampling") k words in the corpus from a α-smoothed unigram distribution:

$$P(c) = \frac{f(c)^\alpha}{\sum_{c'} f(c')^\alpha}. \quad (2)$$

Here, $f(c)$ represents the frequency of the word c and the parameter $\alpha \in \mathbb{R}$ smoothes the distribution. An α equal to 1 leads to a sampling based on the frequency distribution, an α equal to 0 makes items being sampled equally, while a negative α makes unpopular items being sampled more often than popular onces. The parameter α is empirically set to 0.75 following [17]. Better performance and faster training can be obtained by using a dynamic window-size (i.e.: randomly sampling the window size between 1 and L) or by randomly removing words (sub-sampling) that are more frequent than some threshold t with a probability

$$p(c) = \frac{f(c) - t}{f(c)} - \sqrt{\frac{t}{f(c)}}. \quad (3)$$

In recommendation settings, such as music consumption or online shopping, a revised version of the Distributional Hypothesis is adopted to justify the use of SGNS, stating that items that appear in the same contexts share similarities. Grbovic et al. [8] proposed the use of SGNS on sequences of items to form item embeddings employed in recommendation applications. W2V-based item embeddings have since been successfully used in numerous recommendation scenarios [1, 18, 20, 26]. Since then, this method has been derived to handle problems specific to recommendation. For example, Meta-Prod2vec [26], improves upon Prod2vec by using the item meta-data side information to regularize the final item embedding, and authors show that they outperforms Prod2vec on NEP for music, globally and especially in a cold-start regime.

2.2 Hyperparameters

In the following, we describe the role and classically used values of the hyperparameters in the investigated literature [1, 8, 18–20, 26], for which simultaneous optimization significantly improved NEP performance.

Negative pairs of items (w, c_N) are sampled from the negative sampling distribution which is parametrized by α in Equation (2). The original smoothed unigram distribution, proposed in [17], samples items proportionally to their frequency raised to the power $\alpha = 0.75$. This value was empirically chosen because it outperformed the uniform ($\alpha = 1$) and unigram ($\alpha = 0$) distributions on every linguistic task tested by the authors. This result was further confirmed in [15], where the authors extensively studied the marginal effect of optimizing each hyperparameter of W2V. Consequently, widely used implementations of W2V (e.g. Gensim [21]) use this value by default, and does not present it as tunable. We assume that works that do not discuss this parameter rely on its commonly accepted default value.

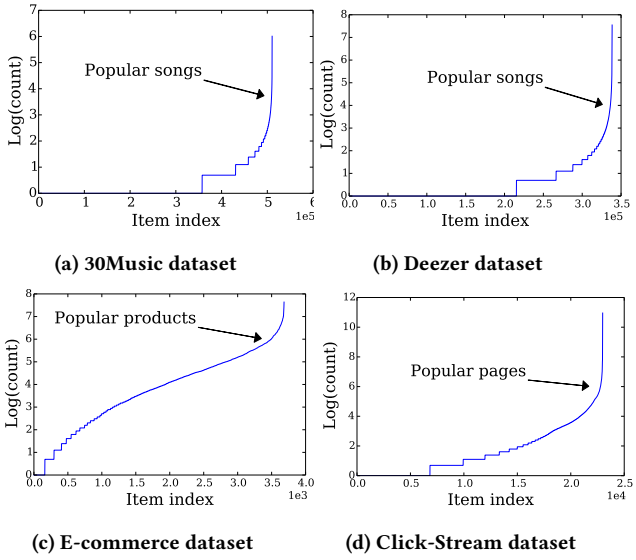
The number of epochs n controls the total number of times SGNS goes over each item of the dataset, which has a direct impact on the duration and the quality of the training. Its default value is set to 5 in Gensim [21]. Some work did hyperparameter search on the number of epochs [20, 26] on a range we extended in this work, or do not develop on the methods nor the final values used.

The window-size is sampled randomly between 1 and the maximum window-size L. It controls how wide the gap between two items in a sequence can be, such that they are still considered in the same context. The default value is set to 5 in Gensim [21]. Some authors claim that it is best to use a "infinite" window-size [1], meaning that the whole sessions is considered as one context, but most arbitrarily use a fixed value without further discussion.

Higher-frequency items are randomly sub-sampled, according to Equation (3). The default value of the parameter t is set to 10^{-3} in Gensim [21]. This variable is hardly discussed in the investigated literature [1, 8], and to our knowledge never optimized.

3 EXPERIMENTS

We study the influence of 7 hyperparameters (including n, L, t, α) on final performance by evaluating SGNS on a recommendation task based on items embeddings, with 4 recommendation datasets coming from diverse sources. Our code is available online. [1]



(a) 30Music dataset (b) Deezer dataset

(c) E-commerce dataset (d) Click-Stream dataset

Figure 1: Log count distributions of the considered datasets.

3.1 Datasets

3.1.1 Music datasets. We rely on 2 sets of listening sessions. The former, "30Music" [25], composed of listening and playlists data retrieved from Internet radio stations, is open and commonly

[1] Code and datasets for reproducing our results can be found at https://github.com/deezer/w2v_reco_hyperparameters_matter

used for recommendation [2, 4, 11, 26]. The latter is a dataset of listening sessions from Deezer, a French on-demand music streaming service. Both are composed of 100k sessions sampled from the original datasets. We refer to these datasets as 30Music and Deezer datasets, respectively. Log count distributions for these datasets are shown in Figure 1. The log count distribution tail is sharp: there is an important discrepancy between popular and unpopular items. We notice a strong resemblance between the two distributions, which suggests similarities of music usage between users of the two platforms.

3.1.2 E-commerce dataset. We use an open Online Retail dataset [6] composed of transactions occurring between 01/12/2010 and 09/12/2011 for a UK-based and registered non-store online retail. It is composed of 4234 user purchase histories. Compared to music data, the log count distribution tail is heavier: the discrepancy between popular and unpopular items is smaller.

3.1.3 Click-stream dataset. We use the "kosarak" dataset [3], which contains anonymized click-stream data of a Hungarian online news portal. It is composed of 83625 user click-stream histories. The log count distribution tail is comparable to the two music datasets.

3.2 Task and metrics

We evaluate the item embeddings on the NEP task, a common way to assess the quality of item embeddings [14, 22, 26] for recommendation. We consider time ordered sequences of user interactions with the items. We split each sequence into training, validation and test sets. We first fit the SGNS model on the first $(n-1)$ elements of each user sequence; then, we use the performance on randomly sampled $((n-1)$-th, n-th) pairs of items (validation set) to bench the hyperparameters, and finally, we report our final results by performing prediction on randomly sampled $((n-1)$-th, n-th) pairs of items (test set, disjoint with validation set). For prediction, we use the last item in the training sequence as the query item and predict the k closest items to the query item using a nearest-neighbor approach [7]. We use 10k test/validation pairs for 30Music, Deezer and Click-Stream datasets and 2k for E-Commerce dataset. We evaluate with the following metrics:

- Hit ratio at K (HR@K). It is equal to 1 if the test item appears in the list of k predicted items and 0 otherwise [13].
- Normalized Discounted Cumulative Gain (NDCG@K). It favors higher ranks in the ordered list of predicted items [12]. Since only one of the k retrieved items can be relevant (i.e. equal to the n-th item in the sequence), the formula writes

$$NDGC@K = \begin{cases} \frac{1}{log_2(j+1)} & \text{if } j^{th} \text{ predicted item is correct} \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

3.3 Experimental setup

We use a modified implementation of Gensim [21] for our experiments, such that parameter α (Eq. (1)) becomes tunable. We perform a hyperparameter search (300k models evaluated) on: the number of epochs n (10 to 200 with step of +10), the window-size L (3, 7, 12, 15), the sub-sampling parameter t (Eq. (2)) (10^{-5} to 10^{-1} with step of $\times 10$), the negative sampling distribution parameter α (Eq. (3)) (−1.4 to 1.4 with step of +0.2), ~~the embedding size (50 to 200 with a step of 50), the number of negative samples (5 to 20 with a step of 5) and the learning rate (0.0025 to 0.25 with a step of ×10)~~. The marginal benefit of including the 3 latter variables to the optimization is not significant, with less than 2% in terms of performance. Thus, for readability, we only focus on the influence of the 4 first hyperparameters and keep the other fixed to default values (respectively 50, 5 and 0.025).

We run the task on the 4 datasets described in Section 3.1 and select the optimal parameters based on the HR@10 performance, given that we observe a strong correlation with NDCG@10 performance. Results (average score over 10 folds) and 95% confidence intervals on the test set are aggregated in Table 1 ("Fully Optimized SGNS"). To demonstrate the benefit of performing a hyperparameter search, we present the results obtained when SGNS is used with default values as defined in Gensim [21] implementation in the "Out-of-the-box SGNS" row. As parameter α is often not tunable in implementations and never discussed in the recommendation setting, we also report the results obtained when optimizing on every hyperparameter but α in the "*Optimized* SGNS" row, in order to isolate the benefit of optimizating over this variable.

To compare the benefit of using recommendation-specific implementations of Prod2vec, we use Meta-Prod2vec [26] on the two music datasets, having artists as side information, and report results in the "Fully optimized Meta-Prod2vec" row for optimized models, and "Meta-Prod2vec [26]" for models trained with the configuration specified in [26]. As Meta-Prod2vec was specifically designed to perform well on a cold-start regime, we also report, in Table 2, results on the cold-start scenario, for pair of (query item, next item) that have zero or less than 3 co-occurrences in the training set.

4 RESULTS

On the two Music datasets, performing a hyperparameter search roughly doubles the performance (Table 1), over using the default values. The best configurations for these two datasets are quasi-identical (same α, sub-sampling parameters and window-size), which is possibly a consequence of the observed similarity of count distributions in Section 3. Hyperparameter optimization allows to increase performance by a factor of 10 for the Click-Stream dataset, and yields substantial performance gains for the E-commerce dataset.

Interestingly, for all datasets, the marginal benefit of including α in the hyperparameter search is significant in terms of final performance. This is illustrated in Figure 2, where we select the best performing configurations for different α values and plot the NEP performance for 30Music dataset. The original $\alpha = 0.75$, optimal on linguistic tasks, is clearly suboptimal on this recommendation setting. The optimal hyperparameter α is -0.5, such that the optimal negative sampling distribution is one more likely to sample unpopular items as negative examples. For the Deezer and Click-Stream datasets, the optimal α is also negative.

We observe that Meta-Prod2vec [26] can also benefit from a hyperparameter optimization, with once again a negative α. However, we also note that it is outperformed by an optimized SGNS

Table 1: Next Event Prediction results for music, e-commerce and click-stream datasets with different hyperparameter optimization strategy. The 4 specified hyperparameters are respectively: window-size, number of epochs, sub-sampling parameter and negative sampling distribution parameter.

Model	30Music dataset (HR@10)	30Music dataset (NDCG@10)	Deezer dataset (HR@10)	Deezer dataset (NDCG@10)	E-commerce dataset (HR@10)	E-commerce dataset (NDCG@10)	Click-stream dataset (HR@10)	Click-stream dataset (NDCG@10)
Out-of-the-box SGNS (hyperparameters: L, n, t, α)	11.16 ± 0.1 ($5,5,10^{-3}$,0.75)	0.099 ± 0.001 ($5,5,10^{-3}$,0.75)	8.13 ± 0.1 ($5,5,10^{-3}$,0.75)	0.061 ± 0.004 ($5,5,10^{-3}$,0.75)	22.21 ± 0.1 ($5,5,10^{-3}$,0.75)	0.159 ± 0.001 ($5,5,10^{-3}$,0.75)	3.07 ± 0.1 ($5,5,10^{-3}$,0.75)	0.018± 0.001 ($5,5,10^{-3}$,0.75)
Optimized SGNS (hyperparameters: L, n, t, α)	22.24 ± 0.1 ($3,90,10^{-5}$,0.75)	0.166 ± 0.001 ($3,90,10^{-5}$,0.75)	14.43 ± 0.1 ($3,90,10^{-5}$,0.75)	0.100 ± 0.001 ($3,90,10^{-5}$,0.75)	26.17 ± 0.1 ($3,140,10^{-3}$,0.75)	0.181 ± 0.001 ($3,140,10^{-3}$,0.75)	24.14 ± 0.5 ($7,150,10^{-5}$,0.75)	0.130 ± 0.003 ($7,150,10^{-5}$,0.75)
Fully optimized SGNS (hyperparameters: L, n, t, α)	**23.75** ± 0.1 ($3,110,10^{-5}$,-0.5)	**0.174** ± 0.001 ($3,110,10^{-5}$,-0.5)	**15.73** ± 0.1 ($3,130,10^{-5}$,-0.5)	**0.108** ± 0.001 ($3,130,10^{-5}$,-0.5)	**26.34** ± 0.1 ($3,140,10^{-3}$,1)	**0.183** ± 0.001 ($3,140,10^{-3}$,1)	**26.26** ± 0.2 ($7,150,10^{-5}$,-1)	**0.147** ± 0.002 ($7,150,10^{-5}$,-1)
MetaProd2vec [26] (hyperparameters: L, n, t, α)	19.41 ± 0.2 ($3, 10, 10^{-3}$, 0.75)	0.142 ± 0.001 ($3, 10, 10^{-3}$, 0.75)	14.24 ± 0.1 ($3, 10, 10^{-3}$, 0.75)	0.097 ± 0.001 ($3, 10, 10^{-3}$, 0.75)	-	-	-	-
Fully optimized MetaProd2vec (hyperparameters: L, n, t, α)	20.85 ± 0.1 ($7, 90, 10^{-4}$, −0.5)	0.152 ± 0.001 ($7, 90, 10^{-4}$, −0.5)	**15.62** ± 0.1 ($3, 150, 10^{-4}$, −0.5)	**0.108** ± 0.001 ($3, 150, 10^{-4}$, −0.5)	-	-	-	-

Table 2: NEP performance (HR@10) in cold-start regime as a function of training frequency of the pair (query item, next item) on 30Music and Deezer datasets.

Model (dataset)	Pair frequency = 0	Pair frequency < 3
Fully optimized SGNS (30Music) (hyperparameters: L, n, t, α)	8.29 ± 0.1 ($3,110,10^{-5}$,-0.5)	**16.48** ± 0.1 ($3,110,10^{-5}$,-0.5)
Fully optimized MetaProd2vec (30Music) (hyperparameters: L, n, t, α)	**8.84** ± 0.1 ($7, 90, 10^{-4}, -0.5$)	15.79 ± 0.1 ($7, 90, 10^{-4}, -0.5$)
Fully optimized SGNS (Deezer) (hyperparameters: L, n, t, α)	4.52 ± 0.1 ($3,130,10^{-5}$,-0.5)	**9.81** ± 0.1 ($3,130,10^{-5}$,-0.5)
Fully optimized MetaProd2vec (Deezer) (hyperparameters: L, n, t, α)	**5.43** ± 0.1 ($3, 150, 10^{-4}, -0.5$)	**9.98** ± 0.1 ($3, 150, 10^{-4}, -0.5$)

on 30Music and on par on Deezer dataset (Table 1). On the cold-start regime, results indicates that, once optimized, MetaProd2vec is on par with SGNS (Table 2). Hence, it might be worth optimizing standard methods before moving to more specialized methods.

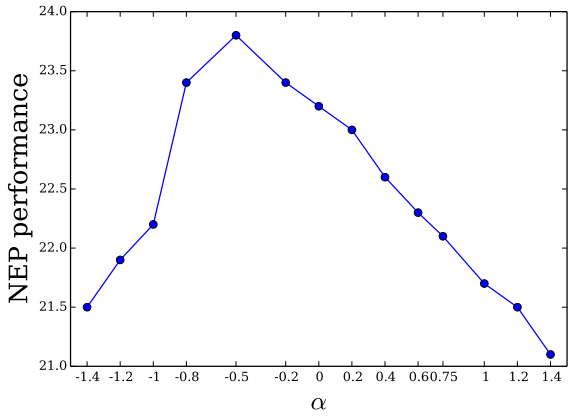


Figure 2: Best final performance of SGNS as a function of negative sampling parameter α for 30Music dataset.

Results confirm that the optimal choice hyperparameters for SGNS are data-dependent and task-dependent, and that, for the given datasets and the considered task (NEP), it is highly valuable in terms of final performance to simultaneously optimize the hyperparameters. Especially, the optimal negative sampling distribution clearly differs from the one proven to be optimal for linguistic tasks [15, 17], and optimizing over this additional variable yields significant improvements. As a negative α leads to sample more often unpopular items for negative samples, and positive samples are necessarily more popular by construction, the algorithm is made to better distinguish between items of different order of popularity. This correlates with the observation that items tend to share the same order of popularity within a music streaming session.

5 CONCLUSION

Developed first for NLP, SNGS generate words embeddings that help achieving state of the art performance in semantic similarity and analogy tasks. Previous work shows that it can be directly applied to sequences of items to generate item embeddings useful for recommendations. Interestingly, while NLP data and tasks differ in their structure and goals from those of recommendation, the hypothesis behind some of the parameters of the algorithms are barely discussed, nor their default fixed values revised. We show that using different values for some hyperparameters, namely negative sampling distribution, number of epochs, subsampling parameter and window-size, leads to significantly better performances on classical evaluation tasks on 4 recommendation datasets. While performing a hyperparameter search for each different types of data and tasks in a real-life recommendation setting can be time consuming and computationally costly, we stress out the benefits of having better item embeddings to better distinguish, cluster and classify content, which can lead to substantial gains in demanding industries, such as on-demand music streaming services, where a few bad recommendations can quickly lead a user to leave the service.

Comparing several recommendation datasets on the same task, we observe that different data distributions result in different optimal hyperparameter values. The homogeneity of popularity between items of a same sequence, the shape of the popularity distribution, or the heterogeneity of the items in the catalog have a direct impact on the task evaluation. We have yet to find if and how we can induce the optimal hyperparameter values from the structure of the data. This could have a strong impact on improving the current results with SNGS applied to recommendation.

REFERENCES

[1] Oren Barkan and Noam Koenigstein. 2016. Item2vec: neural item embedding for collaborative filtering. In *Machine Learning for Signal Processing (MLSP), 2016 IEEE 26th International Workshop on*. IEEE, 1–6.

[2] Shay Ben-Elazar, Gal Lavee, Noam Koenigstein, Oren Barkan, Hilik Berezin, Ulrich Paquet, and Tal Zaccai. 2017. Groove Radio: A Bayesian Hierarchical Model for Personalized Playlist Generation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 445–453.

[3] Ferenc Bodon. 2003. A fast APRIORI implementation.. In *Frequent Itemset Mining Implementations Workshop, 2003. Third IEEE International Conference on Data Mining (ICDM)*. IEEE.

[4] Mattia Brusamento, Roberto Pagano, MA Larson, and Paolo Cremonesi. 2016. Explicit Elimination of Similarity Blocking for Session-based Recommendation. *RecSys 2016 Poster Proceedings* (2016).

[5] Joan L Bybee and Paul J Hopper. 2001. *Frequency and the emergence of linguistic structure*. Vol. 45. John Benjamins Publishing.

[6] Daqing Chen, Sai Laing Sain, and Kun Guo. 2012. Data mining for the online retail industry: A case study of RFM model-based customer segmentation using data mining. *Journal of Database Marketing & Customer Strategy Management* 19, 3 (2012), 197–208.

[7] Thomas Cover and Peter Hart. 1967. Nearest neighbor pattern classification. *IEEE transactions on information theory* 13, 1 (1967), 21–27.

[8] Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan, and Doug Sharp. 2015. E-commerce in your inbox: Product recommendations at scale. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1809–1818.

[9] R Douglas Greer, Laura G Dorow, Gustav Wachhaus, and Elmer R White. 1973. Adult approval and students' music selection behavior. *Journal of Research in Music Education* 21, 4 (1973), 345–354.

[10] Frank Hutter, Holger Hoos, and Kevin Leyton-Brown. 2014. An efficient approach for assessing hyperparameter importance. In *International Conference on Machine Learning*. 754–762.

[11] Dietmar Jannach, Iman Kamehkhosh, and Lukas Lerche. 2017. Leveraging multi-dimensional user models for personalized next-track music recommendation. In *Proceedings of the Symposium on Applied Computing*. ACM, 1635–1642.

[12] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[13] Quoc V Le, Alex Smola, Olivier Chapelle, Choon Hui Teo, and Ralf Herbrich. 2007. Direct optimization of ranking measures. *KDD'13*.

[14] Benjamin Letham, Cynthia Rudin, and David Madigan. 2013. Sequential event prediction. *Machine learning* 93, 2-3 (2013), 357–380.

[15] Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics* 3 (2015), 211–225.

[16] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[17] 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.

[18] Cataldo Musto, Giovanni Semeraro, Marco De Gemmis, and Pasquale Lops. 2015. Word Embedding Techniques for Content-based Recommender Systems: An Empirical Evaluation. *RecSys 2015 Poster Proceedings*.

[19] Thomas Nedelec, Elena Smirnova, and Flavian Vasile. 2016. Content2vec: Specializing joint representations of product images and text for the task of product recommendation. (2016).

[20] Makbule Gulcin Ozsoy. 2016. From word embeddings to item recommendation. *arXiv preprint arXiv:1601.01356* (2016).

[21] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en.

[22] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*. ACM, 811–820.

[23] Magnus Sahlgren. 2008. The distributional hypothesis. *Italian Journal of Disability Studies* 20 (2008), 33–53.

[24] Harald Steck. 2011. Item popularity and recommendation accuracy. In *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 125–132.

[25] Roberto Turrin, Massimo Quadrana, Andrea Condorelli, Roberto Pagano, and Paolo Cremonesi. 2015. 30Music Listening and Playlists Dataset.. In *RecSys 2015 Poster Proceedings*.

[26] Flavian Vasile, Elena Smirnova, and Alexis Conneau. 2016. Meta-Prod2Vec: Product Embeddings Using Side-Information for Recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 225–232.