Semantic interpretability of latent factors for recommendation*

Vito Walter Anelli, Tommaso Di Noia, Eugenio Di Sciascio, Claudio Pomo
Polytechnic University of Bari
Bari, Italy
firstname.lastname@poliba.it

Azzurra Ragone
Independent Researcher
Milan, Italy
azzurra.ragone@gmail.com

ABSTRACT

Model-based approaches to recommendation have proven to be very accurate. Unfortunately, exploiting a latent space we miss references to the actual semantics of recommended items. In this extended abstract, we show how to initialize latent factors in Factorization Machines by using semantic features coming from a knowledge graph in order to train an interpretable model. Finally, we introduce and evaluate the semantic accuracy and robustness for the knowledge-aware interpretability of the model.

1 INTRODUCTION

Transparency and interpretability of predictive models are gaining momentum since they have been recognized as a key element in the next generation of recommendation algorithms. When equipped with interpretability of recommendation results, a system ceases to be just a black-box and users are more willing to extensively exploit the predictions [6]. However, powerful and accurate Deep Learning or model-based recommendation algorithms and techniques project items and users in a new vector space of latent features thus making the final result not directly interpretable. In the last years, many approaches have been proposed that take advantage of side information to enhance the performance of latent factor models. Interestingly, in [7] the authors argue about a new generation of knowledge-aware recommendation engines able to exploit information encoded in knowledge graphs $\mathcal{KG}$ to produce meaningful recommendations. In this work, we propose a *knowledge-aware Hybrid Factorization Machine* (kaHFM) to train interpretable models in recommendation scenarios taking advantage of semantics-aware information. kaHFM relies on Factorization Machines (*FM*) [4] and it extends them in different key aspects by making use of the semantic information encoded in a knowledge graph. We show how kaHFM may exploit data coming from knowledge graphs as side information to build a recommender system whose final results are accurate and, at the same time, semantically interpretable.

2 KNOWLEDGE-AWARE HYBRID FACTORIZATION MACHINES

In [1], the authors proposed to encode a Linked Data knowledge graph in a Vector Space Model (*VSM*) to develop a Content Based recommender system. Given a set of items $I = \{i_1, i_2, \dots, i_N\}$ in a catalog and their associated triples $\langle i, \rho, \omega \rangle$ in a knowledge graph $\mathcal{KG}$, we may build the set of all possible features as $F = \{\langle \rho, \omega \rangle \mid \langle i, \rho, \omega \rangle \in \mathcal{KG} \text{ with } i \in I\}$. Each item can be then represented as a vector of weights $\mathbf{i} = [v_{(i,\langle\rho,\omega\rangle_1)}, \dots, v_{(i,\langle\rho,\omega\rangle_{|F|})}]$, where $v_{(i,\langle\rho,\omega\rangle)}$ is the generic element computed as the normalized TF-IDF value for $\langle \rho, \omega \rangle$. Since the numerator of $TF^{\mathcal{KG}}$ can only take values 0 or 1 and, each feature under the root in the denominator has value 0 or 1, $v_{(i,\langle\rho,\omega\rangle)}$ is zero if $\langle \rho, \omega \rangle \notin \mathcal{KG}$, and otherwise:

$$v_{(i,\langle\rho,\omega\rangle)} = \frac{\log|I| - \log|\langle j, \rho, \omega\rangle \cap \mathcal{KG} | j \in I|}{\sqrt{\sum\limits_{\langle\rho,\omega\rangle\in F} |\{\langle\rho,\omega\rangle \mid \langle i,\rho,\omega\rangle \in \mathcal{KG}\}|}} \tag{1}$$

Analogously, when we have a set U of users, we may represent them using the features describing the items they enjoyed in the past. We use f to denote a feature $\langle \rho, \omega \rangle \in F$. Given a user u, if we denote with I^u the set of the items enjoyed by u, we may introduce the vector $\mathbf{u} = [v_{(u,f_1)} \cdots, v_{(u,f_{|F|})}]$, where $v_{(u,f)}$ is the generic element computed as:

$$v_{(u,f)} = \frac{\sum\limits_{i \in I^u} v_{(i,f)}}{|\{i \mid i \in I^u \text{ and } v_{(i,f)} \neq 0\}|}$$

Given the vectors $\mathbf{u}_j$, with $j \in [1 \dots |U|]$, and $\mathbf{i}_p$, with $p \in [1 \dots |I|]$, we build a matrix $\mathbf{V} \in \mathbb{R}^{n \times |F|}$, where $n = |U| + |I|$: so the first $|U|$ rows have a one to one mapping with $\mathbf{u}_j$ while the last ones correspond to $\mathbf{i}_p$. In second degree Factorization Machines models the score is computed as:

$$\hat{y}(\mathbf{x}^{\mathbf{ui}}) = w_0 + \sum_{j=1}^{n} w_j \cdot x_j + \sum_{j=1}^{n} \sum_{p=j+1}^{n} x_j \cdot x_p \cdot \sum_{f=1}^{k} v_{(j,f)} \cdot v_{(p,f)} \tag{2}$$

We may see that, for each $\mathbf{x}$, the term $\sum_{j=1}^{n} \sum_{p=j+1}^{n} x_j \cdot x_p \cdot \sum_{f=1}^{k} v_{(j,f)} \cdot v_{(p,f)}$ is non-zero, i.e., when both x_j and x_p are equal to 1. In a recommendation scenario, this happens when there is an interaction between a user and an item. Moreover, the summation $\sum_{f=1}^{k} v_{(j,f)} \cdot v_{(p,f)}$ represents the dot product between two vectors: $\mathbf{v}_j$ and $\mathbf{v}_p$ with a size equal to k. Hence, $\mathbf{v}_j$ represents a latent representation of a user, $\mathbf{v}_p$ that of an item within the same latent space, and their interaction is evaluated through their dot product.

In order to inject the knowledge coming from $\mathcal{KG}$ into kaHFM, we set $k = |F|$ in Equation 2. In other words, we impose a number of latent factors equal to the number of features describing all the items in our catalog. Since we formulated our problem as a *top-N* recommendation task, kaHFM can be trained using a learning to rank approach like Bayesian Personalized Ranking Criterion (BPR)[5] obtaining $\hat{\mathbf{V}}$. We extract the items vectors $\mathbf{v}_j$ from $\hat{\mathbf{V}}$, and we use them to implement an Item-kNN recommendation approach. We measure similarities between each pair of items i and j by evaluating the cosine similarity of their corresponding vectors in $\hat{\mathbf{V}}$. In an RDF knowledge graph, we usually find different types of encoded information. We extracted the categorical information that is mainly used to state something about the subject of an entity.

*An extended version of this work will be presented at the *International Semantic Web Conference (ISWC 2019)*[2]

IIR 2019, September 16–18, 2019, Padova, Italy

3 EXPERIMENTAL EVALUATION

We evaluated the performance of our method on two well-known datasets for recommender systems belonging to movies domain: Yahoo!Movies[1][2], and Facebook Movies[3]. Experiments were conducted adopting the "All Unrated Items" protocol, and an Hold-Out 80-20 temporal split [3]. All the items from the datasets come with a DBpedia link. We retrieved all the $\langle \rho, \omega \rangle$ pairs[4] excluding some noisy features (based on the following predicates): owl:sameAs, dbo:thumbnail, foaf:depiction, prov:wasDerivedFrom, foaf:isPrimaryTopicOf.

Accuracy Evaluation. The goal of this evaluation is to assess if the controlled injection of Linked Data positively affects the training of *FM*. We compared kaHFM[5] w.r.t. a canonical 2 degree *FM* optimized via BPR (BPR-FM). In order to preserve the expressiveness of the model, we used the same number of hidden factors as kaHFM. Since we use items similarity in the last step of our approach, we compared kaHFM against an *Attribute Based Item-kNN* (ABItem-kNN) algorithm, where each item is represented as a vector of weights, computed through a TF-IDF model. We also compared kaHFM against Item-kNN, and User-kNN based on Cosine Similarity, Most-Popular, and a knowledge-graph-based *VSM* adopting the representation formalted in [1]. To evaluate our approach, we measured accuracy through *Precision@N*, and Normalized Discounted Cumulative Gain (*nDCG@N*). Table 1 shows the corresponding

	Facebook Movies	Yahoo!Movies	
Categorical Setting (CS)	Precision@10	Precision@10	nDCG@10
ABItem-kNN	0.0173*	<u>0.0421</u>*	<u>0.1174</u>*
BPR-FM	0.0158*	0.0189*	0.0344*
MostPopular	0.0118*	0.0154*	0.0271*
ItemKnn	<u>0.0262</u>*	0.0203*	0.0427*
UserKnn	0.0168*	0.0231*	0.0474*
VSM	0.0185*	0.0385*	0.1129*
kaHFM	**0.0296**	**0.0524**	**0.1399**

Table 1: Accuracy results for Facebook Movies, and Yahoo!Movies considering Top-10 recommendations, and a relevance threshold of 4 over 5 stars.

results. We highlight in **bold** the best result while we <u>underline</u> the second one. Statistically significant differences in performance are denoted with a * mark considering Student's paired t-test with a 0.05 level.

Semantic Accuracy. The main idea behind Semantic Accuracy is to evaluate, given an item i, how well kaHFM is able to return its original features available in the computed top-K list $\mathbf{v}_i$. In other words, subset i represented by $F^i = \{f_1^i, \ldots, f_m^i, \ldots f_M^i\}$, with $F^i \subseteq F$, we check if the values in $\mathbf{v}_i$, corresponding to $f_{m,i} \in F^i$, are higher than those corresponding to $f \notin F^i$. For the set of M features initially describing i we see how many of them appear in the set $top(\mathbf{v}_i, M)$ representing the top-M features in $\mathbf{v}_i$. We then normalize this number by the size of F^i and average on all the items within the catalog I. Table 2 shows the results for SA@nM with $n \in \{1, 2, 3, 4, 5\}$ and $M = 10$, and evaluated the number of ground features available in the top-nM elements of $\mathbf{v}_i$ for each dataset.

Generative Robustness. To check if kaHFM promotes important features for an item i we proposed a new measure: *Generative*

[1]Yahoo! Webscope dataset ydata-ymovies-user-movie-ratings-content-v1_0

[2]http://research.yahoo.com/Academic_Relations

[3]https://2015.eswc-conferences.org/program/semwebeval.html

[4]https://github.com/sisinflab/LinkedDatasets/

[5]https://github.com/sisinflab/HybridFactorizationMachines/

Semantics Accuracy	SA@M	SA@2M	SA@3M	SA@4M	SA@5M	F.A.
Yahoo!Movies	0.847	0.863	0.865	0.868	0.873	12.143
Facebook Movies	0.864	0.883	0.889	0.894	0.899	12.856

Table 2: Semantics Accuracy results for different values of M. F.A. denotes the Feature Average number per item.

Robustness. We suppose that a particular feature $\langle \rho, \omega \rangle$ is useful to describe an item i but the corresponding triple $\langle i, \rho, \omega \rangle$ is not represented in the knowledge graph. In case kaHFM was robust in generating weights for unknown features, it should discover the importance of that feature and modify its value to make it enter the Top-K features in $\mathbf{v}_i$. Starting from this observation, the idea to measure robustness is then to "forget" a triple involving i and check if kaHFM can generate it. Given a catalog I, we may then define the *Robustness for 1 removed feature @M* (1-Rob@M) as the number of items for which the removed feature is in $Top - M$ after training. Similarly to SA@nM, we may define 1-Rob@nM. Table 2

1-Robustness	1-Rob@M	1-Rob@2M	1-Rob@3M	1-Rob@4M	1-Rob@5M	F.A.
Yahoo!Movies	0.487	0.645	0.713	0.756	0.793	12.143
Facebook Movies	0.821	0.945	0.970	0.980	0.984	12.856

Table 3: 1-Robustness for different values of M. Column F.A. denotes the Feature Average number per item.

showed that kaHFM was able to guess 10 on 12 different features for Yahoo!Movies. In this experiment, we remove one of the ten features (thus, based on Table 2, kaHFM will guess an average of $10 - 1 = 9$ features). Since the number of features is 12 we have 3 remaining "slots". In Table 3, we measure how often kaHFM is able to guess the removed feature in these "slots".

4 CONCLUSION AND FUTURE WORK

We have proposed an interpretable method for recommendation scenario, kaHFM, in which we bind the meaning of latent factors for a Factorization machine to data coming from a knowledge graph. We considered Categorical information coming from DBpedia and we have shown that the generated recommendations are more precise and personalized on two different publicly available datasets. We showed that the computed features are semantically meaningful, and the model is robust regarding computed features. In the future we want to test the kaHFM performance in classical Information Retrieval, and knowledge graph completion tasks.

REFERENCES

[1] Vito Walter Anelli, Tommaso Di Noia, Pasquale Lops, and Eugenio Di Sciascio. 2017. Feature Factorization for Top-N Recommendation: From Item Rating to Features Relevance. In *Proc. of the 1st Workshop on Intelligent Recommender Systems by Knowledge Transfer & Learning co-located with ACM Conf. on Recommender Systems (RecSys 2017), Como, Italy, August 27, 2017. (CEUR Workshop Proceedings)*, Vol. 1887. CEUR-WS.org, 16–21.

[2] Vito Walter Anelli, Tommaso Di Noia, Eugenio Di Sciascio, Azzurra Ragone, and Joseph Trotta. 2019. How to make latent factors interpretable by feeding Factorization machines with knowledge graphs. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, NZ, October 26-30, 2019.*

[3] Vito Walter Anelli, Tommaso Di Noia, Eugenio Di Sciascio, Azzurra Ragone, and Joseph Trotta. 2019. Local Popularity and Time in top-N Recommendation. In *Advances in Information Retrieval - 41st European Conference on IR Research, ECIR 2019, Cologne, Germany, April 14-18, 2019, Proceedings, Part I.* 861–868.

[4] Steffen Rendle. 2010. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th Int. Conf. on.* IEEE, 995–1000.

[5] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI 2009, Proc. of the Twenty-Fifth Conf. on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009.* 452–461.

[6] Markus Zanker. 2012. The influence of knowledgeable explanations on users' perception of a recommender system. In *Sixth ACM Conf. on Recommender Systems, RecSys '12, Dublin, Ireland, September 9-13, 2012.* 269–272.

[7] Yongfeng Zhang and Xu Chen. 2018. Explainable Recommendation: A Survey and New Perspectives. *CoRR* abs/1804.11192 (2018).