Moving From Item Rating to Features Relevance in Top-N Recommendation

Vito Walter Anelli[1], Tommaso Di Noia[1], Eugenio Di Sciascio[1], Pasquale Lops[2], and Joseph Trotta[1]

[1] Polytechnic University of Bari, Via E. Orabona, 4, Bai, Italy
{vitowalter.anelli,tommaso.dinoia,eugenio.disciascio,joseph.trotta}@poliba.it
[2] University of Bari "Aldo Moro", Via E. Orabona, 4, Bai, Italy
pasquale.lops@uniba.it

Abstract. Although very effective in computing accurate recommendations, due to their inner nature, collaborative algorithms work very well with dense matrices but show their limits when they deal with sparse ones. In these cases, using only past ratings may lead to unsatisfactory results in the recommendation list. In this paper we show how to move from a user-item to a user-feature matrix by exploiting original user ratings. We then use matrix factorization techniques to compute recommendations.

1 Introduction

Matrix factorization techniques have proven their effectiveness in improving the performance of recommendation engines in a pure collaborative approach and are implemented in many industrial and commercial systems [2]. Whenever available, descriptions of the items can be used as a valuable source of information to augment the knowledge injected in and exploited by the system to compute the recommendation list of items. More recently, thanks to the Linking Open Data initiative, many structured data have become freely available to represent the content of items in different knowledge domains and then feed recommendation engines [3]. Several works have tried to build recommender systems by exploiting Linked Open Data (LOD) as side information for representing items, in addition to the user preferences usually collected through ratings. Properties gathered from `DBpedia`, the cornerstone dataset of the LOD cloud, may be used in different ways: (1) to define semantic similarity measures for providing more accurate recommendations [8, 4]; (2) to deal with problems as the *limited content analysis* or *cold-start*, e.g. by introducing new relevant features to improve item representations [10], or to cope with the increasing data sparsity [5]; (3) to provide a good balance between different recommendation objectives, such as

An extended version of this paper has been published at [1]

IIR 2018, May 28-30, 2018, Rome, Italy.

accuracy and diversity [5]. In [7], for instance, effective strategies to incorporate item features for *top-N* recommender systems are developed. Recently, an interesting approach called Feature Preferences Matrix Factorization (FPMF) has been proposed in [6]. FPMF incorporates user feature preferences in a matrix factorization to predict user likes. It is worth to note that the previously mentioned approaches does not rely on features coming from the `Linked Open Data` cloud. Features composing the description of an item, whatever the source, are not considered per se in the recommendation process but are usually exploited to evaluate the similarity between items or users. We believe that more attention should be paid to modeling the recommendation problem with a focus on recommending features rather then items. Expanding an item in its features brings with it some interesting side effects. On the one hand, all features may represent relations that, e.g., latent factor models we are not able to look at. On the other hand, features give us a new set of explicit connections between items to be exploited with collaborative filtering algorithms. Finally, recommending items via feature recommendation may lead to an easier generation of explanations for the recommended list of items. Unfortunately, moving from items to features is not that straight as in a forest of many features, most of them may result not relevant to a user. Moreover, once we design an algorithm able to compute a recommendation list of features, we have to go back to the items space, as the ultimate goal of a recommender systems is to suggest items to a user. In this paper we present `FF` (for `Features Factorization`), a *top-N* recommendation algorithm originally introduced in [1] that relies on user's feature preferences and collaborative filtering information in the features space. The main goal of `FF` is to compute an ordered list of features preferred by the user and, starting from such list, to reassemble the relevance values of each returned feature to produce a *top-N* list of items to recommend. All the side information adopted by `FF` with reference to a specific item i is retrieved from `DBpedia` in form of triples $\langle i, p, e \rangle$. For each item in the user profile we retrieve its features by querying `DBpedia` thus getting them as a set of entities e.

The remainder of the paper is structured as follows. In the next section we introduce and describe `FF`. We than close the paper with a section devoted to Conclusion and future works.

2 Proposed Approach

Motivation. This work aims at investigating the role of feature **rating** and **relevance** in the item rating process. The main intuition behind `FF` is that items can be handled as a collection of features on which the recommendation process is then performed. If we want to discover the contribution of each single feature in the evaluation, first of all, we need to unpack each item in its composing features. Then, by combining the overall popularity of each feature in the user profile (feature relevance) and the rating assigned to items containing that feature we may estimate the implicit rating the user is giving to that specific feature. The second observation we based our work on, is that the relevance of an item in

the user profile cannot be entirely encoded in its ratings as the single rating represents a degree of liking about the specific item.

Data Model. Each item in the user profile is associated with a relevance function we denote with $\rho^{ui}(\cdot)$. Its value represents an estimation of how important is a particular item to the user u. Analogously, we have a value associated to each feature in the profile computed via the function $\rho^{uf}(\cdot)$ computing the relevance of the feature f in the user profile. Actually, each feature is associated also with a rating $r^{uf}(\cdot)$ which is inferred by considering the rating of all the items containing f.

Problem Formulation. By considering the data associated to the user profile as described in the previous section we can move from a rating matrix connecting user and items to a user-feature matrix where each value is represented by the pair $\langle \rho^{uf}(\cdot), r^{uf}(\cdot) \rangle$. In other words, we may consider two user-feature matrices: the one $\mathcal{P}$ containing relevance values $\rho^{uf}(\cdot)$, the other $\mathcal{R}$ including the inferred ratings $r^{uf}(\cdot)$.

In `FF`, the relevance of a feature pe is computed as its probability of belonging to the set I_u representing the items already rated by a user u. More formally we have:

$$\rho^{uf}(pe) = \frac{\sum_{i \in I_u} |\{\langle i, p, e \rangle \mid \langle i, p, e \rangle \in \texttt{DBpedia}\}|}{|I_u|}$$

The idea behind this computation is quite straight: the more a feature is connected to the items in the user profile, the higher its relevance for the user.

Once we have computed the relevance of all the features in the user profile, we can move to the computation of the relevance for the items $i \in I_u$. This can be computed as the normalized summation of the relevance for all the features it is composed by. In formulas, we have

$$\rho^{ui}(i) = \frac{\sum_{\langle i,p,e \rangle \in \texttt{DBpedia}} \rho^{uf}(pe)}{|\{\langle i, p, e \rangle \mid \langle i, p, e \rangle \in \texttt{DBpedia}\}|}$$

Given a feature pe, the computation of the feature rating $r^{uf}(pe)$ exploits both the rating and the relevance of each item $i \in I_u$ containing pe.

$$r^{uf}(pe) = \frac{\sum_{\langle i,p,e \rangle \in \texttt{DBpedia}} r_{ui} \cdot \rho^{ui}(i)}{\sum_{\langle i,p,e \rangle \in \texttt{DBpedia}} \rho^{ui}(i)} \tag{1}$$

top-N Recommendation. The profiles we built contain only the features the user met before, but usually the number of those features is dramatically smaller than the overall number of features and this results in $\mathcal{P}$ and $\mathcal{R}$ being very sparse. In order to complete the information they contain, we compute, via Biased Matrix Factorization, the missing values $\hat{\rho}^{uf}(pe)$ for $\mathcal{P}$ and $\hat{r}^{uf}(pe)$ for $\mathcal{R}$. We run matrix factorization independently on $\mathcal{P}$ and $\mathcal{R}$. $\hat{\rho}^{uf}(pe)$ and $\hat{r}^{uf}(pe)$ represent the predicted relevance and the predicted rating for all those features not belonging to any of the items in I_u. As the resulting matrices contain both content-based and collaborative informations (due to the matrix factorization), we refer to them as *hybrid profile*.

With the *hybrid profile* we can estimate a ranked list for all the remaining items within the collection. In fact, the ranking of an item in the list is computed by considering the rating of the features belonging to the item and their relevance.

$$\hat{r}^{ui}(i) = \sum_{(\langle i,p,e\rangle \in \texttt{DBpedia}) \wedge (i \in I_u)} \rho^{uf}(pe) \cdot r^{uf}(pe) + \sum_{(\langle i,p,e\rangle \in \texttt{DBpedia}) \wedge (i \notin I_u)} \hat{\rho}^{uf}(pe) \cdot \hat{r}^{uf}(pe)$$

It is important to point out that these estimations do not correspond to an actual rating but the correct item ranking is yet preserved. In order to improve the results of the final recommendation process, we may reduce the number of features considered while computing the final rank based on their relevance and popularity [1].

3 Conclusion and Future Works

In this paper we presented `FF`, a novel algorithm that bases on feature recommendation as an intermediate step for computing *top-N* items recommendation lists. The main idea behind `FF` is that feature relevance in a user profile plays a key role in the selection and rating of an item in a collection. As future work, we are investigating the behavior of `FF` with respect to novelty and diversity of results. We are also interested in exploring the behavior of `FF` approach with different collaborative filtering algorithms, other than factorization techniques in the item-feature space and in particular with Factorization Machines [9].

References

1. Anelli, V.W., Di Noia, T., Lops, P., Di Sciascio, E.: Feature factorization for top-n recommendation: From item rating to features relevance. In: Proceedings of the 1st Workshop on Intelligent Recommender Systems by Knowledge Transfer & Learning co-located with ACM Conference on Recommender Systems (RecSys 2017), Como, Italy, August 27, 2017. pp. 16–21 (2017)
2. Bell, R.M., Koren, Y.: Lessons from the netflix prize challenge. Acm Sigkdd Explorations Newsletter **9**(2), 75–79 (2007)
3. Di Noia, T., Mirizzi, R., Ostuni, V.C., Romito, D., Zanker, M.: Linked open data to support content-based recommender systems. In: Proceedings of the 8th International Conference on Semantic Systems. pp. 1–8. ACM (2012)
4. Di Noia, T., Ostuni, V.C., Rosati, J., Tomeo, P., Di Sciascio, E., Mirizzi, R., Bartolini, C.: Building a relatedness graph from linked open data: A case study in the IT domain. Expert Syst. Appl. **44**, 354–366 (2016). https://doi.org/10.1016/j.eswa.2015.08.038, https://doi.org/10.1016/j.eswa.2015.08.038
5. Musto, C., Basile, P., Lops, P., de Gemmis, M., Semeraro, G.: Introducing linked open data in graph-based recommender systems. Information Processing & Management **53**(2), 405–435 (2017)
6. Nasery, M., Braunhofer, M., Ricci, F.: Recommendations with optimal combination of feature-based and item-based preferences. In: Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization, UMAP. pp. 269–273 (2016)
7. Ning, X., Karypis, G.: Sparse linear methods with side information for top-n recommendations. In: Sixth ACM Conference on Recommender Systems, RecSys. pp. 155–162 (2012)
8. Piao, G., Breslin, J.G.: Measuring semantic distance for linked open data-enabled recommender systems. In: Proceedings of the 31st Annual ACM Symposium on Applied Computing. pp. 315–320 (2016)
9. Rendle, S.: Factorization machines. In: Proceedings of the 2010 IEEE International Conference on Data Mining. pp. 995–1000. ICDM '10, IEEE Computer Society, Washington, DC, USA (2010). https://doi.org/10.1109/ICDM.2010.127, http://dx.doi.org/10.1109/ICDM.2010.127
10. Schmachtenberg, M., Strufe, T., Paulheim, H.: Enhancing a location-based recommendation system by enrichment with structured data from the web. In: 4th International Conference on Web Intelligence, Mining and Semantics WIMS. pp. 17:1–17:12 (2014)