Deriving item features relevance from collaborative domain knowledge

Maurizio Ferrari Dacrema
Politecnico di Milano
maurizio.ferrari@polimi.it

Alberto Gasparin
Universitá della Svizzera Italiana
alberto.gasparin@usi.ch

Paolo Cremonesi
Politecnico di Milano
paolo.cremonesi@polimi.it

ABSTRACT

An Item based recommender system works by computing a similarity between items, which can exploit past user interactions (collaborative filtering) or item features (content based filtering). Collaborative algorithms have been proven to achieve better recommendation quality then content based algorithms in a variety of scenarios, being more effective in modeling user behaviour. However, they can not be applied when items have no interactions at all, i.e. *cold start items*. Content based algorithms, which are applicable to cold start items, often require a lot of feature engineering in order to generate useful recommendations. This issue is specifically relevant as the content descriptors become large and heterogeneous. The focus of this paper is on how to use a collaborative models domain-specific knowledge to build a wrapper feature weighting method which embeds collaborative knowledge in a content based algorithm. We present a comparative study for different state of the art algorithms and present a more general model. This machine learning approach to feature weighting shows promising results and high flexibility.

ACM Reference Format:
Maurizio Ferrari Dacrema, Alberto Gasparin, and Paolo Cremonesi. 2018. Deriving item features relevance from collaborative domain knowledge. In *Proceedings of KaRS 2018 Workshop on Knowledge-aware and Conversational Recommender Systems (KaRS @RecSys 2018).* ACM, New York, NY, USA, 4 pages.

1 INTRODUCTION

Recommender systems aim at guiding the user through the navigation of vast catalogs and in recent years they have become widespread. Among item based algorithms, content based are the most widely used, as they provide good performance and explainability. Content Based algorithms recommend items based on similarities computed via item attributes. Although being applicable in any circumstance in which at least some information about the items is available, they suffer from many drawbacks related to the quality of the item features. It is hard and expensive to provide an accurate and exhaustive description of the item. In recent years the amount of data which is machine-readable on the web has increased substantially, it is therefore possible to build heterogeneous and complex representations for each item (e.g. textual features extracted from web pages). Those representations however often comprise of high number of features and are sparse and noisy, to the point where adding new data hampers the recommendation quality. Feature weighting, which can be considered as a generalization of feature selection, is a useful tool to improve content based algorithms. Traditional Information Retrieval methods like TF-IDF and BM25 [10], while often leading to accuracy improvements, cannot take into account how important are those features from the user point of view. Collaborative Filtering on the other hand determines items similarity by taking into account the user interactions. It is known that collaborative filtering generally outperforms content based filtering even when few ratings for each user are available [5]. The main disadvantage of collaborative systems is their inability to compute predictions for new items or new users due to the lack of interactions, this problem is referred to as *cold-start* item. In cold start scenarios only content based algorithm are applicable, this is the case in which improving item description would be most beneficial, and is therefore the main focus of this article.

The most common approach to tackle the cold-start item problem is to rely on content-based algorithms, whose accuracy is sometimes much poorer. In this paper we further investigate the cold-start item problem focusing on how to learn feature weights able to better represent feature importance from the user point of view. We provide a comparative study of state of the art algorithms and present a more general model demonstrating its applicability on a wide range of collaborative models. Moreover we describe a two step approach that:

(1) exploits the capability of a generic collaborative algorithms to model domain-specific user behaviour and achieve state-of-the-art performance for warm items
(2) embeds the collaborative knowledge into feature weights

The rest of the paper is organized as follows. In Section 2 we briefly review the literature in the cold-start recommendation domain, in Section 3 our framework is presented and a comparison of the different algorithms is discussed in Section 4. Finally conclusions and future works are highlighted in Section 5

2 RELATED WORKS

Various tools are at our disposal to assess the relevance of a feature. We can distinguish feature weighting algorithms in three categories: filtering, embedding and wrappers [4].

Filtering methods usually rely on information retrieval. Methods like TF-IDF or BM25 are not optimized with respect to a predictive model, therefore the resulting weights are not domain-specific and can not take into account the rich collaborative information, even when available.

Embedding methods learn feature weights as a part of the model training, examples of this are UFSM [3] and FBSM [11]. Among embedded methods main drawbacks are a complex training phase and

KaRS @RecSys 2018, October 7, 2018, Vancouver, Canada.

noise sensitivity due to the strong coupling of features and interactions. *User-Specific Feature-based Similarity Models* (UFSM) learns a personalized linear combination of similarity functions known as global similarity functions and can be considered as a special case of Factorization Machines. *Factorized Bilinear Similarity Models* (FBSM) was proposed as an evolution of UFSM and aims to discover relations among item features. The FBSM similarity matrix is computed as follows:

$$sim(i,j) = \mathbf{f}_i^T \mathbf{W} \mathbf{f}_j \tag{1}$$

where f is the feature vector of the item, W is the matrix of parameters whose diagonal elements represents how well a feature of item i interacts with the same feature of item j, while the off diagonal elements determines the correlation among different features. In order to reduce the number of parameters W is represented as the summation of diagonal weights and a low rank approximation of the off-diagonal values:

$$\mathbf{W} = \mathbf{D} + \mathbf{V}^T \mathbf{V} \tag{2}$$

where D is a diagonal matrix having as dimension the number of features, n_F, and $V \in \mathbb{R}^{n_L \times n_F}$. The number of latent factors n_L is treated as a parameter.

Wrapper methods rely on a two step approach, by learning feature weights on top of an already available model, an example of this is *Least-square Feature Weights* (LFW) [1]. LFW learns feature weights from a SLIM similarity matrix using a simpler model with respect to FBSM.

$$sim(i,j) = \mathbf{f}_i^T \mathbf{D} \mathbf{f}_j \tag{3}$$

All these algorithms, to our best knowledge, have never been subject to a comparative study.

3 SIMILARITY BASED FEATURE WEIGHTING

Feature weighting can be formulated as a minimization problem whose objective function is:

$$\underset{\mathbf{W}}{\operatorname{argmin}} \quad \left\| \mathbf{S}^{(\mathbf{CF})} - \mathbf{S}^{(\mathbf{W})} \right\|_F^2 + \lambda \left\| \mathbf{D} \right\|_F^2 + \beta \left\| \mathbf{V} \right\|_F^2 \tag{4}$$

where $\mathbf{S}^{(\mathbf{CF})}$ is any item-item collaborative similarity, $\mathbf{S}^{(\mathbf{W})}$ is the similarity function described in Equation (1), $\mathbf{W}$ is the feature weight matrix which captures the relationships between items features and β and λ are the regularization terms, we call this model *Collaborative boosted Feature Weighting* (CFW). This model can either use the latent factors (CFW D+V), as in FBSM, or not (CFW D), as in LFW.

The advantages of learning from a similarity matrix, instead of using the user interactions, are several:

- High flexibility in choosing the collaborative algorithm which can be treated as a black box
- Similarity values are less noisy than user interactions
- The model is simpler and convergence is faster

In this paper a two steps hybrid method is presented, in order to easily allow to embed domain-specific user behaviour, as represented by a collaborative model, in a weighted content based recommender. The presented model is easily extendable to other algorithms and domains. The learning phase is composed by two steps. The goal of the first step is to find the optimal parameters for the collaborative algorithm, to this end a collaborative algorithm is trained and tuned on warm items. The second step applies an embedded method to learn the optimal item feature weights that better approximate the item-item collaborative similarity obtained before.

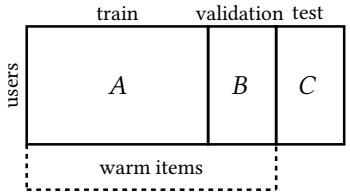

Figure 1: URM split, A and B contains the warm items while C contains the cold items.

3.1 Parameter estimation

We solve (4) via SGD applying Adam [6] which is well suited for problems with noisy and sparse gradients. Note that here our goal is to find weights that will approximate as well as possible the collaborative similarity, this is why we optimize MSE and not BPR, which was used in FBSM. The objective function is therefore:

$$\mathcal{L}_{MSE}(D,V) = \frac{1}{2} \sum_{i \in I} \sum_{j \in I} \left(\hat{s}_{ij} - s_{ij}^{CF} \right)^2 + \lambda \left\| \mathbf{D} \right\|_F^2 + \beta \left\| \mathbf{V} \right\|_F^2 \tag{5}$$

4 EVALUATION

We performed experiments to confirm that our approach is capable of embedding collaborative knowledge in a content based algorithm improving its recommendation quality in an item cold-start scenario.

4.1 Dataset

In order to evaluate our approach we used only item descriptors accessible via web, which we assume will be available for new items, excluding user generated content. The datasets are the following:

Netflix. Enriched with structured and unstructured attributes extracted from IMDB. This dataset has 250k users, 6.5k movies, 51k features and 8.8M ratings in 1-5 scale. The rating data is enriched with 6.6k binary editorial attributes such as director, actor and genres.

The Movies Database. [1]: 45k movies with 190k TMDB editorial features and ratings for 270k users. This dataset has been built from the original one by extracting its 70-cores.

For all the listed datasets, features belonging to less than 5 items or more than 30% of the items have been removed, as done in [11].

4.2 Evaluation procedure

The evaluation procedure consists of two steps.

Step 1 - Collaborative algorithm: In this step the training of the collaborative algorithm is performed on warm items, which

[1] https://www.kaggle.com/rounakbanik/the-movies-dataset

Algorithm		Netflix Precision	Recall	MRR	MAP	NDCG	The Movies Precision	Recall	MRR	MAP	NDCG
Content baselines	CBF KNN	0.0439	0.0405	0.1177	0.0390	0.0449	0.3885	0.0916	0.6909	0.3166	0.1641
	CBF KNN IDF	0.0439	0.0405	0.1177	0.0390	0.0449	0.3931	0.0930	0.6956	0.3215	0.1662
	CBF KNN BM25	0.0466	0.0410	0.1237	0.0414	0.0462	0.3931	0.0930	0.6956	0.3215	0.1662
hybrid feature weights baselines	FBSM	0.0244	0.0240	0.0476	0.0162	0.0199	0.2957	0.0727	0.5503	0.2114	0.1192
	LFW	0.0679	0.0573	0.1632	0.0631	0.0646	0.4135	0.0959	0.7073	0.3442	0.1736
CFW - D	CF KNN	0.0688	0.0597	0.1585	0.0609	0.0645	0.3891	0.0906	0.6939	0.3192	0.1597
	P3alpha	**0.0714**	**0.0679**	**0.1707**	**0.0664**	**0.0716**	0.3847	0.0882	0.6911	0.3179	0.1578
	RP3beta	0.0656	0.0624	0.1643	0.0610	0.0669	**0.4281**	**0.1010**	**0.7233**	**0.3588**	**0.1806**
	SLIM RMSE	0.0643	0.0529	0.1572	0.0583	0.0604	0.4058	0.0923	0.7017	0.3372	0.1669
	SLIM BPR	0.0685	0.0539	0.1583	0.0618	0.0619	0.4170	0.0967	0.7218	0.3455	0.1723

Table 1: Performance of CFW and baselines evaluated on cold items.

are defined as the union of split A and B, see Figure 1. Hyper-parameters are tuned with a 20% holdout validation.

Step 2 - Feature weights: In this case a cold-item validation is chosen as it better represents the end goal of the algorithm to find weights that perform well on cold items. The collaborative similarity will be learned using only split A with the hyper-parameters found in the previous step, see Figure 1. An embedded method is then used to learn the optimal item feature weights that better approximate the item-item collaborative similarity obtained before. The hyper-parameters of the machine learning model are tuned using split B while set C is used for pure testing.

4.3 Collaborative Similarity

The collaborative similarity matrices used in our approach are computed using different algorithms: KNN collaborative (using the best-performing similarity among: Cosine, Pearson, Adjusted Cosine, Jaccard), P3alpha [2] (graph based algorithm which models a random walk), Rp3beta [9] (reranked version of P3alpha), SLIM BPR [8] and SLIM MSE [7].

4.4 Results

Table 1 shows the recommendation quality of both pure content based and hybrid baselines, as well as CFW D evaluated on all collaborative similarity models. Table 2 shows the performance of the two components of CFW and FBSM on Netflix, results for the other dataset are omitted as they behave in the same way.

From Table 1 we can see that FBSM performs poorly, which indicates that while it has the power to model complex relations, it is more sensitive to noise and data sparsity than other algorithms. Learning from a similarity matrix, as LFW and CFW D+V does, results in much better results than FBSM. In Table 2 it is possible to see that the latent factor component was able to learn very little, this suggests that while rendering the model more expressive, it introduces noise and numerical instability. Note that the performance using of the diagonal component alone is higher than the one obtained by adding the V component. However, its effectiveness could be influenced by the feature structure and therefore it might be relevant in some specific cases. rom Table 1 we can see

Model		Precision	Recall	MRR	MAP	NDCG
FBSM	D + V	0.0244	0.0240	0.0476	0.0162	0.0199
	D	**0.0348**	**0.0366**	**0.0954**	**0.0312**	**0.0379**
	V	0.0138	0.0112	0.0247	0.0071	0.0086
CFW	D + V	0.0475	0.0424	0.1336	0.0430	0.0489
	D	**0.0635**	**0.0579**	**0.1653**	**0.0602**	**0.0641**
	V	0.0412	0.0346	0.1146	0.0335	0.0392

Table 2: Model component contribution on the result of FBSM and CFW on Netflix.

that by using only the diagonal and discarding the latent factor component, the performance improves significantly.

While LFW only learned feature weights using a SLIM similarity matrix, our results indicate that it is possible to learn from a wide variety of item based algorithms, even those not relying on machine learning. This means that machine learning feature weights can be used on top of already available collaborative algorithm with little effort. Using an intermediate similarity matrix while offering additional degrees of freedom in the selection of the collaborative model, also simplifies the training phase and improves overall performance.

5 CONCLUSIONS

In this paper we presented different state of the art feature weighting methods, compared their performance and proposed a more general framework to effectively apply machine learning feature weighting to boost content based algorithms recommendation quality, embedding user domain-specific behaviour. We also demonstrate high flexibility in the choice of which collaborative algorithm to use. Future work directions include testing the proposed approach in different datasets and domains, as well as exploring the symmetric problem of using the collaborative similarity to discover item features or to reduce feature noise.

REFERENCES

[1] Leonardo Cella, Stefano Cereda, Massimo Quadrana, and Paolo Cremonesi. 2017. Deriving Item Features Relevance from Past User Interactions. In *Proceedings of*

the 25th Conference on User Modeling, Adaptation and Personalization. ACM, 275–279.

[2] Colin Cooper, Sang Hyuk Lee, Tomasz Radzik, and Yiannis Siantos. 2014. Random walks in recommender systems: exact computation and simulations. In *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 811–816.

[3] Asmaa Elbadrawy and George Karypis. 2015. User-Specific Feature-Based Similarity Models for Top-n Recommendation of New Items. *ACM Trans. Intell. Syst. Technol.* 6, 3, Article 33 (April 2015), 20 pages. DOI: http://dx.doi.org/10.1145/2700495

[4] Isabelle Guyon and André Elisseeff. 2003. An introduction to variable and feature selection. *Journal of machine learning research* 3, Mar (2003), 1157–1182.

[5] Domonkos Tikk IstvÃąn PilÃąszy. 2009. Recommending new movies: even a few ratings are more valuable than metadata. *RecSys 09 Proceedings of the third ACM conference on Recommender systems* (2009), 93 – 100. DOI: http://dx.doi.org/10.1145/1639714.1639731

[6] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[7] Mark Levy and Kris Jack. 2013. Efficient top-n recommendation by linear regression. In *RecSys Large Scale Recommender Systems Workshop*.

[8] Xia Ning and George Karypis. 2011. Slim: Sparse linear methods for top-n recommender systems. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*. IEEE, 497–506.

[9] Bibek Paudel, Fabian Christoffel, Chris Newell, and Abraham Bernstein. 2017. Updatable, Accurate, Diverse, and Scalable Recommendations for Interactive Applications. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 7, 1 (2017), 1.

[10] Stephen Robertson, Hugo Zaragoza, and Michael Taylor. 2004. Simple BM25 extension to multiple weighted fields. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*. ACM, 42–49.

[11] Mohit Sharma, Jiayu Zhou, Junling Hu, and George Karypis. 2015. Feature-based factorized bilinear similarity model for cold-start top-n item recommendation. In *Proceedings of the 2015 SIAM International Conference on Data Mining*. SIAM, 190–198.

This figure "map.png" is available in "png" format from:

http://arxiv.org/ps/1811.01905v1

This figure "sigchi-logo.png" is available in "png" format from:

http://arxiv.org/ps/1811.01905v1