

# A Joint Framework for Collaborative and Content Filtering

Justin Basilico  
Department of Computer Science  
Brown University  
Providence, RI, USA  
basilico@cs.brown.edu

Thomas Hofmann  
Department of Computer Science  
Brown University  
Providence, RI, USA  
th@cs.brown.edu

## ABSTRACT

This paper proposes a novel, unified, and systematic approach to combine collaborative and content-based filtering for ranking and user preference prediction. The framework incorporates all available information by coupling together multiple learning problems and using a suitable kernel or similarity function between user-item pairs. We propose and evaluate an on-line algorithm (JRank) that generalizes perceptron learning using this framework and shows significant improvement over other approaches.

## Categories and Subject Descriptors

H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—*Information Filtering*

## 1. INTRODUCTION

The prediction of user ratings and preferences when interacting with a computer system is a key challenge for applications such as information filtering and recommender systems. A standard approach to rating prediction is known as *collaborative filtering* [4] where correlations across a database of users are used to make a prediction for the active user. An alternative paradigm is *content-based filtering* where content descriptors of the item are used to predict the rating. We pursue the philosophy that collaborative and content-based filtering are complementary views that should be unified in a common learning architecture. Formally, we follow an approach first suggested in [1] which is to learn a mapping from user-item pairs to a set of ratings.

## 2. JOINT FEATURE MAPS AND KERNELS

To state our modeling approach more formally, we denote by  $\mathcal{U}$  a set of users and by  $\mathcal{X}$  a set of items. A joint feature map is a mapping  $\Psi : \mathcal{U} \times \mathcal{X} \rightarrow \mathbb{R}^D$  which extracts  $D$  features from user-item pairs. We define a family of functions  $F$  that are linear in the chosen feature map via  $F(u, x; w) = \langle \Psi(u, x), w \rangle$ , where  $w \in \mathbb{R}^D$  is a weight vector. To predict ratings we operate in the setting of *ordinal regression* and use a set of adaptive thresholds  $\theta$  to quantize  $F$  into bins. If there are  $k$  response levels, then there will be thresholds  $\theta_j \in \mathbb{R}$  with  $1 \leq j \leq k-1$  and  $\theta_k = +\infty$ . The prediction function simply picks the number of the bin the computed  $F$ -value falls into,  $f(u, x; w, \theta) = \min\{j \in \{1, \dots, k\} : F(u, x; w) < \theta_j\}$ .

As we will show, the estimation of  $w$  and  $\theta$  with the proposed algorithm only depends on inner products between feature vectors for user-item pairs with an observed rating. This means that instead of specifying  $\Psi$  explicitly, we can define kernel functions  $K$  over  $\mathcal{U} \times \mathcal{X}$  which define a joint feature map implicitly, i.e. via  $K((u, x), (u', x')) \equiv \langle \Psi(u, x), \Psi(u', x') \rangle$ . This is a convenient and effective way of designing appropriate representations over user-item pairs.

## 2.1 Joint Feature Maps via Tensor Products

In this paper, we restrict ourselves to joint feature maps that are generically constructed from feature maps for items and users via the tensor product. By this we mean a two-stage process of first defining  $\Lambda : \mathcal{U} \rightarrow \mathbb{R}^G$  and  $\Phi : \mathcal{X} \rightarrow \mathbb{R}^H$  and then combining every dimension of  $\Lambda$  multiplicatively with every dimension of  $\Phi$  to get  $\Psi(u, x) = \Lambda(u) \otimes \Phi(x) \in \mathbb{R}^D$  where  $D = G \cdot H$ . This has apparent advantage in the design of joint feature maps, but also yields computational benefits, since we can compute the inner product as  $\langle \Psi(u, x), \Psi(u', x') \rangle = \langle \Lambda(u), \Lambda(u') \rangle \langle \Phi(x), \Phi(x') \rangle$ . The latter implies that we may design kernel functions  $K_U$  for users and  $K_X$  for items independently and combine them multiplicatively to define a joint kernel.

## 2.2 Designing Kernels

We define kernels for users and items that additively combine four elementary kernel functions, which are then combined multiplicatively to yield the joint kernel function.

The simplest kernel function is the diagonal kernel matrix corresponding to the *identity* features ( $K^{\text{id}}$ ). Another type of kernel function can be built from an explicit *attribute* ( $K^{\text{at}}$ ) representation for items or users. For users these attributes may correspond to demographic information such as gender, age, nationality, location, or income. For items such as documents this may encode a standard tf-idf vector space representation, whereas for movies it may include attributes such as genre, cast, crew, or plot synopsis. The third kernel we propose, the *correlation* kernel ( $K^{\text{co}}$ ), encodes collaborative information by defining a kernel matrix  $C$  based on a correlation measure between users. The feature representation of such a kernel would correspond to the (normalized) ratings of the user, though some care has to be taken with missing ratings. A fourth kernel can also be based on the correlation kernel by squaring it. This gives a *quadratic correlation* kernel ( $K^{\text{qu}}$ ) that creates a positive semi-definite matrix  $C^2$ , which can get rid of issues where the correlation matrix  $C$  is not positive-definite or the values are unreliable since correlations some between users have very few items in common. Similar kernels can be devised

								Mean average error		Mean zero-one error		Expected rank utility	
$K_{\mathcal{U}}^{\text{id}}$	$K_{\mathcal{U}}^{\text{at}}$	$K_{\mathcal{U}}^{\text{co}}$	$K_{\mathcal{U}}^{\text{qu}}$	$K_{\mathcal{X}}^{\text{id}}$	$K_{\mathcal{X}}^{\text{at}}$	$K_{\mathcal{X}}^{\text{co}}$	$K_{\mathcal{X}}^{\text{qu}}$	PRank	JRank	PRank	JRank	PRank	JRank
○		○	○				○	1.210	0.880	0.657	<b>0.621</b>	0.739	0.791
○		○	○	○			○	1.211	<b>0.877</b>	0.658	<b>0.621</b>	0.736	<b>0.793</b>
○	○	○	○	○	○		○	1.360	0.882	0.685	0.624	0.709	0.792
Pearson								0.936		0.673		0.736	

Table 1: Results for 100 trials of 100 training users, 5000 input users, and 1000 training/input items.

for items by interchanging the role of user and item.

### 3. PERCEPTRON ALGORITHM

While most previous machine learning approaches decouple the learning problems associated with each user, our approach leads to a joint problem which couples learning across different users and items. In order to deal with the increase in complexity in a reasonable manner we want an on-line algorithm that uses an *ordinal scale*, to avoid interpreting a ranking as an absolute value, and has a dual representation, in order to use kernel functions. The combination of these three criteria leads us to generalize the perceptron ranking algorithm (PRank) of [3] to do joint learning, which we call JRank. We also use a more aggressive update rule than PRank which enforces a margin  $\gamma$  around each threshold.

### 4. EXPERIMENTAL SETUP

We used the EachMovie<sup>1</sup> data set of movie ratings to evaluate our approach. The Internet Movie Database<sup>2</sup> was used to collect item attributes relating to genre, cast, crew, country, language, and keywords of a movie. The plot synopsis was utilized to generate a tf-idf representation. We also used the demographic information about users in EachMovie about gender, age, and residence.

The randomized generation of training and test data is conducted by randomly subsampling rows of the ratings matrix for a given number of users. Then the items (columns) are divided into a training and testing set to produce the submatrices of user-item pairs. We also make use of users as input information beyond those used to generate the training examples. To show the competitiveness of our approach, we compare JRank with a standard collaborative filtering algorithm based on the Pearson correlation coefficient [4] and with single user PRank. Three different evaluation metrics are used: the mean average error, zero-one test error, and the expected rank utility metric of [2].

### 5. RESULTS AND DISCUSSION

In our evaluation of JRank we have systematically investigated various combinations of the four types of features described above. The results indicate the identity features are very useful when the same set of users and/or items used in training are used in testing. The collaborative information of the correlation and quadratic correlation kernels are also very useful, although there is only a minor improvement in combining the two. The least useful features are the ones that encode attributes, although the item attributes can be useful when little collaborative information is given. The user attributes are somewhat of a hindrance because they

encode so little information, but in an application where more user attributes are available they could be useful. Our experiments also show that JRank converges to near optimal after a mere five iterations according to all three error metrics, it always has better performance than PRank, and does better than Pearson after only three iterations.

In evaluating the performance with regards to the amount of input data we have found that when more collaborative information given as input the performance of JRank improves when collaborative kernels are utilized. In addition, using more items for training also improves performance unilaterally. Increasing the number of training users, which is the number of users coupled together in learning, increased performance up until about 50 or 100 users at which point no additional gain is really seen. Table 1 shows the results of some good kernel combinations.

### 6. CONCLUSION

We have presented a learning architecture for the problem of predicting ratings that can incorporate all available information based on the idea of defining joint kernel functions over user-item pairs. We have shown that using this method of coupling together learning problems can achieve substantial improvement in performance over state-of-the-art methods that treat each user individually.

### 7. ACKNOWLEDGMENTS

This work was sponsored by an NSF-ITR grant, award number IIS-0312401.

### 8. REFERENCES

- [1] C. Basu, H. Hirsh, and W. W. Cohen. Recommendation as classification: Using social and content-based information in recommendation. In *Proceedings of the 15th National Conference on Artificial Intelligence*, pages 714–720, 1998.
- [2] J. S. Breese, D. Heckerman, and C. Kardie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.
- [3] K. Crammer and Y. Singer. Pranking with ranking. In *Advances in Neural Information Processing Systems 14*, pages 641–647, 2002.
- [4] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: An open architecture for collaborative filtering of netnews. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, pages 175–186, 1994.

<sup>1</sup>courtesy of Digital Equipment Corporation

<sup>2</sup><http://www.imdb.com>