Mixture-of-interests models for representing users with diverse interests

Maciej Kula
maciej.kula@gmail.com

ABSTRACT

Most existing recommendation approaches implicitly treat user tastes as unimodal, resulting in an average-of-tastes representations when multiple distinct interests are present. We show that appropriately modelling the multi-faceted nature of user tastes through a mixture-of-tastes model leads to large increases in recommendation quality at a modest cost in model complexity. Our result holds both for deep sequence-based and traditional factorization models.

KEYWORDS

Recommender Systems, Matrix Factorization, Sequence-based Recommendations

1 INTRODUCTION

Latent vector-based recommender models commonly represent users with a single latent vector per user [3, 4]. This representation, while simple in formulation and efficient to compute, neglects te fact that users have diverse sub-tastes, and that observed user interactions can be seen as manifestations of several *distinct* tastes or intents. These may either be stable facets of the user's preference (liking both horror and documentary movies, or both bluegrass and classical music), context-driven changes (preference for short-form TV content during the week but long-form cinematography during the week-end), or manifestations of phenomena like account sharing, where two (or more) different users, with correspondingly different tastes, share the same user account.

In all these cases, trying to capture the user's taste in a single latent vector is suboptimal. Firstly, it may lead to lack of nuance in representing the user, where a dominant taste may overpower more niche ones. Secondly it may reduce the quality of item representations, especially when it leads to decreasing the separation in the embedding space between groups of items belonging to multiple tastes or genres. For illustration, suppose that documentaries and horror movies are distinct genres, in the sense that most users prefer one or the other: but that there is still a group of users who like both. To represent that group of users, the genres' embeddings will have to be separated less cleanly than they would be if the model could express the concept of multiple tastes. In general, these problems are similar to that of fitting a unimodal distributional model to bimodal data; for Gaussians, the resulting fitted distribution will be a poor model of the data, and the majority of its density mass will coincide with neither of the true modes.

In this paper, we propose and evaluate representing users as mixtures of several distinct tastes, represented by distinct taste vectors. Each of the taste vectors is coupled with an attention vector, describing how competent it is at evaluating any given item. The user's preference is then modelled as a weighted average of all the user's tastes, with the weights given by how relevant each taste is to evaluating a given item.

We apply this model to both traditional implicit feedback factorization models, and to more recent models using recurrent neural networks. In both cases, our mixture-of-interests models handily outperform single-representation models on standard ranking quality metrics. In the case of deep recurrent sequence models, this improvement is achieved at a very modest cost to to the model complexity, making our model a straightforward improvement over existing methods.

2 RELATED WORK

The idea of moving beyond point embeddings has yielded some interesting results, particularly in the natural language modelling domain.

Vilnis and McCallum [8] embed words as Gaussian distributions, rather than point embeddings. This improves the resulting representations in two ways. Firstly, the resulting representation provides a natural way of expressing uncertainty about the learned representation. Secondly, large (and non-spherical) variances aid the representation of polysemous words. In the context of recommendations, Gausian embeddings with large variances could be used to represent users with wide-ranging tastes.

Athiwaratkun and Wilson [1] extend this idea by representing words a mixtures of Gaussians. This allows them to capture polysemy by modelling each word with several distinct, small-variance Gaussian distributions (rather than artificially inflating the variance of a single distribution). This allows much clearer separation of distinct meanings. This work is closest to the approach presented here.

In the recommender system literature, the idea of using multiple embeddings to represent a user's multiple interests is presented in Weston et al. [10]. In this approach, the recommendation score of an item for a given user is given by the *maximum* of the dot products of the item embedding and each of the user's embedding vectors. This obviates the need for explicit modelling of mixture probabilities, which reduces the number of model parameters and makes evaluation more efficient. However, the model is potentially disadvantaged by its inability to model strong *dis*taste for a given class of items.

3 MODEL

Let U_i be a $m \times k$ matrix representing the m tastes of user i, and A_i be a $m \times k$ matrix representing the affinities of each taste for representing particular items. The recommendation score for item j, represented by a $ktimes1$-dimensional embedding vector e_j, is then given by

$$r_{ij} = \sigma\left(A_i e_j\right) \cdot U_i e_j, \tag{1}$$

where σ is the elementwise sigmoid function, $\sigma\left(A_i e_j\right)$ gives the mixture probabilities, and $U_i e_j$ the recommendation scores output by each mixture component. We assume identity variance matrices for all mixture components.

This gives a mixture of tastes, where the contribution of each individual tastes component is weighted by how competent it is in evaluating item j.

How U_i and A_i are obtained differs between the three evaluated models. In the Mixture-LSTM model, U_{it} and A_{it} (now indexed by t, their position in the sequence of interactions) are linear functions of the hidden state of an LSTM layer trained on the user's previous interactions, z_{it}:

$$z_{it} = \text{LSTM}\left(e_{i1}, e_{i2}, \ldots, e_{it}\right). \tag{2}$$

Given z_{it}, the m-th row of U_{it} and A_{it} is given by

$$\begin{aligned} u^m_{it} &= W^U_m z_{it} + B^U_m \\ a^m_{it} &= W^A_m z_{it} + B^A_m, \end{aligned} \tag{3}$$

where W^U_m, W^A_M, B^U_m, and B^A_m are the learned projection matrices. These are common across all users, representing a modest increase in the total number of model parameters. Note that the LSTM network only needs to be run once to obtain the full user representation.

The Projection Mixture factorization model is similar: an embedding z_i is estimated for each user, and the U_i and A_i matrices are obtained via linear projections:

$$\begin{aligned} u^m_{i} &= W^U_m z_{i} + B^U_m \\ a^m_{i} &= W^A_m z_{i} + B^A_m. \end{aligned} \tag{4}$$

This keeps the number of model parameters small at a potential cost to model capacity. In contrast, the Embedding Mixture factorization model embeds U_i and A_i directly. This substantially increases the number of model parameters, but may lead to better accuracy.

In all models, the input and output item embeddings e are tied.

4 EXPERIMENTS

4.1 Datasets

We use the following datasets in out experiments.

1. Movielens 10M: dataset of 10 million movie ratings across 10,000 movies and 72,000 users [2].
2. Goodbooks: dataset of 6 million ratings across 53,000 users and 10,000 most popular books from the Goodbooks online book recommendation and sharing service [11].
3. Amazon: dataset of ratings and reviews gathered from the Amazon online shopping service [5]. After pruning users and items with fewer than 10 ratings, the dataset contains approximately 4 million ratings from 100,000 users over 114,000 items.

We treat all datasets as implicit feedback datasets, where the existence of an edge between a user and an item expresses implicit preference, and the lack of an edge implicit lack of preference.

4.2 Experimental setup

For factorization models, we split the interaction datasets randomly into train, validation, and test sets. We use 80% of interactions for training, and 10% each for validation and testing. We make no effort to ensure that all items and users in the validation and test sets have a minimum number of training interactions. Our results therefore represent partial cold-start conditions.

Table 1: Dataset statistics

Dataset	Users	Items	Interactions	Density
Movielens 10M	69,879	10,678	10,000,054	0.0134
Amazon	100,085	113,997	3,944,862	0.0003
Goodbooks	53,425	10,001	5,976,479	0.0112

For sequence-based models, we order all interactions chronologically, and split the dataset by randomly assigning users into train, validation, and test sets. This means that the train, test, and validation sets are disjoint along the user dimension. For each dataset, we define a maximum interaction sequence length. This is set to 100 for the Goodbooks and Movielens datasets, and 50 for the Amazon dataset, as the interaction sequences in the Amazon dataset are generally shorter. Sequences shorter than the maximum sequence length are padded with zeros. The models are trained by trying to predict the next item that the user interacts with on the basis of all their prior interactions.

We use mean reciprocal rank (MRR) as our measure of model quality. In factorization models, we use the user representations obtained from the training set to construct rankings over items in the test set. In sequence models, we use the last element of the test interaction sequence as the prediction target; the remaining elements are used to compute the user representation.

We experiment with two loss functions:

- Bayesian personalised ranking (BPR, Rendle et al. [7]), and
- adaptive sampling maximum margin loss, following Weston et al. [9].

For both loss functions, for any known positive user-item interaction pair (i, j), we uniformly sample an implicit negative item $k \in S^-$. For BPR, the loss for any such triplet is given by

$$1 - \sigma\left(r_{ij} - r_{ik}\right), \tag{5}$$

where σ denotes the sigmoid function. The adaptive sampling loss is given by

$$\left|1 - r_{ij} + r_{ik}\right|_+ . \tag{6}$$

For any (i, j) pair, if the sampled negative item k results in a zero loss (that is, the desired pairwise ordering is not violated), a new negative item is sampled, up to a maximum number of attempts. This leads the model to perform more gradient updates in areas where its ranking performance is poorest.

Across all of our experiments, the adaptive maximum margin loss consistently outperforms the BPR loss on both baseline and mixture models. We therefore only report results for the adaptive loss.

We perform extensive hyperparameter optimization across both our proposed models and all baselines. Our goal is two-fold. Firstly, we want to mitigate researcher bias, where more care and attention is devoted to the researcher's proposed model, thus unfairly disadvantaging baseline algorithms in a performance comparison. We believe this to be a common phenomenon; its extent is illustrated, in a related domain, by [6], who find that standard LSTM architectures, when properly tuned, outperform more recent algorithms in natural laguage modelling tasks. Secondly, we wish to understand the extent to which the mixture-of-interests models are *fragile*, in the sense of being highly sensitive to hyperparameter choices. Such fragile algorithms are potentially of lesser utility in industry applications, where the engineering cost of tuning in maintaing them may outweigh the accuracy benefits they bring.

We use random search to tune the algorthms used in our experiments. We optimize batch size, number of training epochs, the learning rate, L2 regularization weight, the loss function, and (where appropriate) the number of taste mixture components.

5 RESULTS

5.1 Hyperparameter search

Figure 1 plots the maximum test MRR achieved by each algorithm as a function of the number of elapsed hyperparameter search iterations. Both baseline and mixture models benefit from hyperparameter tuning. All algorithms converge to their optimum performance relatively quickly, suggesting a degree of robustness to hyperparameter choices. Mixture-LSTM and Embedding Mixture models quickly outperform their baseline counterparts, and maintain a stable performance lead thereafter. This lends support to our belief that the mixture models' superior accuracy reflects their greater capacity to model the recommendation problem well, rather than being an artifact of the experimental procedure or researcher bias.

5.2 Number of taste components

We summarize the effect of increasing the number of taste mixture components in Table 3. By and large, there is a dose-response relationship between the number of mixtures and recommendation quality: being able to represents more distinct user tasts yields better results.

Table 2: Experimental results

Model	Movielens 10M	Amazon	Goodbooks
LSTM	0.0901	0.1502	0.1158
Mixture-LSTM	0.0999	0.1889	0.1358
Bilinear	0.0999	0.1001	0.0738
Projection Mixture	0.0983	0.0698	0.0712
Embedding Mixture	0.0952	0.1696	0.0853

[1] Ratio of binary model MRR to real-valued model MRR

[2] Predictions Per Millisecond: how many items can be scored per millisecond

[3] Ratio of binary PPMS to real-valued PPMS

[4] Ratio of memory required to store binary vs. real-valued parameters

Table 3: Effect of number of mixture components

(a) Sequence models

Components	Movielens 10M	Amazon	Goodbooks
2	0.0921	0.1538	0.1262
4	0.0999	0.1689	0.1304
6	0.0991	0.1889	0.1358
8	0.0982	0.1865	0.1327

(b) Factorization models

Components	Movielens 10M	Amazon	Goodbooks
2	0.0952	0.1343	0.0797
4	0.0952	0.1583	0.0840
6	0.0913	0.1612	0.0811
8	0.0907	0.1696	0.0853

REFERENCES

[1] Ben Athiwaratkun and Andrew Gordon Wilson. 2017. Multimodal Word Distributions. *arXiv preprint arXiv:1704.08424* (2017).

[2] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 5, 4 (2016), 19.

[3] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on.* Ieee, 263–272.

[4] Yehuda Koren. 2009. The bellkor solution to the netflix grand prize. *Netflix prize documentation* 81 (2009), 1–10.

[5] Jure Leskovec, Lada A Adamic, and Bernardo A Huberman. 2007. The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)* 1, 1 (2007), 5.

[6] Gábor Melis, Chris Dyer, and Phil Blunsom. 2017. On the State of the Art of Evaluation in Neural Language Models. *arXiv preprint arXiv:1707.05589* (2017).

[7] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence.* AUAI Press, 452–461.

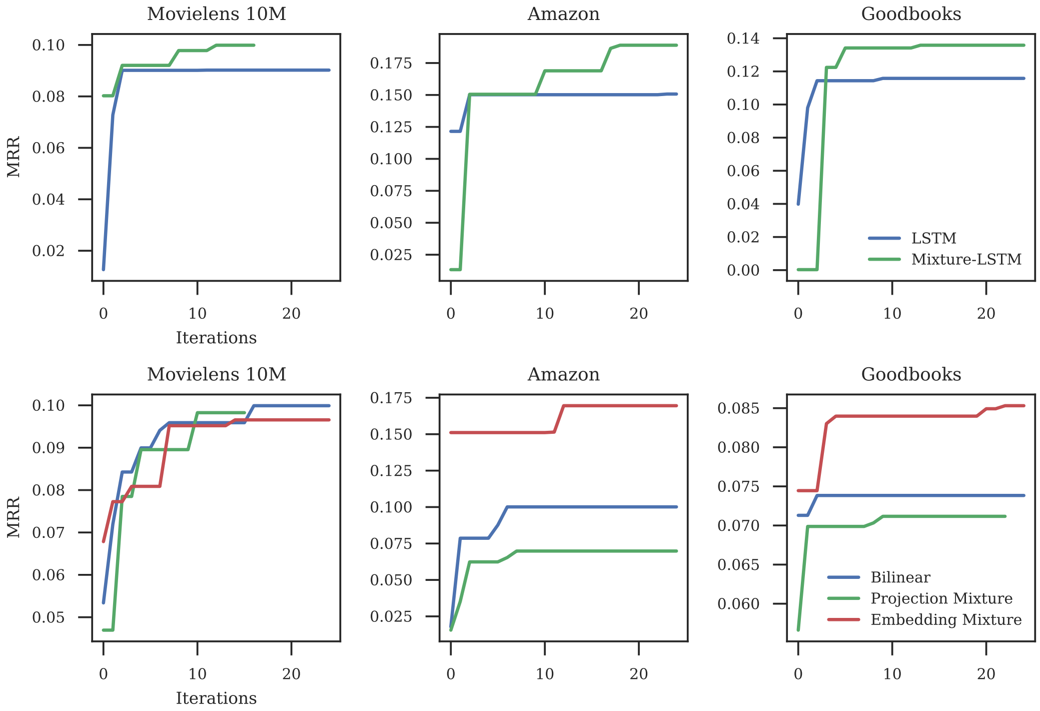
[8] Luke Vilnis and Andrew McCallum. 2014. Word Representations via Gaussian Embedding. *arXiv preprint arXiv:1412.6623* (2014).

[9] Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*, Vol. 11. 2764–2770.

[10] Jason Weston, Ron J Weiss, and Hector Yee. 2013. Nonlinear latent factorization by embedding multiple user interests. In *Proceedings of the 7th ACM conference on Recommender systems.* ACM, 65–68.

[11] Zygmunt Zajac. 2017. Goodbooks-10K dataset. https://github.com/zygmuntz/goodbooks-10k. *GitHub* (2017).

Figure 1: Maximum test MRR vs number of hyperparameter search iterations. Sequence-based models in the top row; factorization-based models in the bottom row.


Movielens 10M
Amazon
Goodbooks
MRR
LSTM
Mixture-LSTM
Iterations
Movielens 10M
Amazon
Goodbooks
MRR
Bilinear
Projection Mixture
Embedding Mixture
Iterations