Mixture-of-tastes 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. Our result holds both for deep sequence-based and traditional factorization models, and is robust to careful selection and tuning of baseline models. In sequence-based models, this improvement is achieved at a very modest cost in model complexity, making mixture-of-tastes models a straightforward improvement on existing baselines.

Introduction

Latent vector-based recommender models commonly represent users with a single latent vector per user (Koren, 2009; Hu, Koren, and Volinsky, 2008). This representation, while simple in formulation and efficient to compute, neglects the 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.

Throughout, we take a take great care to select and carefully tune our baseline models.

Related work

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

Vilnis and McCallum (2014) 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, Gaussian embeddings with large variances could be used to represent users with wide-ranging tastes.

Athiwaratkun and Wilson (2017) 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, Weiss, and Yee (2013). 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.

Model

We propose three variants of the mixture-of-tastes model, covering both recurrent models and traditional factorization models.

Our sequence-based model is the Mixture LSTM (**M-LSTM**) model. It builds on prior work (Wu et al. (2017); Hidasi et al. (2015)) using recurrent neural networks to model the sequence of user interactions. In our model, we use a long-short term memory network (LSTM, Hochreiter and Schmidhuber (1997)) to transform the sequence of user interactions into a latent representation; we then project it into the item embedding space via a number of linear projection layers to obtain the mixture-of-tastes representation. By using a recurrent architecture, we capture both information from both the identity of the items the user interacted with as well as the order in which those interactions occurred.

Within the classical matrix factorization framework, we experiment with two approaches to modelling taste mixtures. The first is the Projection-Mixture Factorization (**PM-F**) model. It extends the basic matrix factorization model by using linear projection layers on top of user latent vectors to model multiple tastes. In contrast, the Embedding-Mixture Factorization (**EM-F**) model directly embeds all the mixture parameters for each user. This makes it less parsimonious, but potenitally more expressive.

Formally, 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 $k \times 1$-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 given by each mixture component. We assume identity variance matrices for all mixture components.

How U_i and A_i are obtained differs between the three evaluated models. In the M-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 z_{it} of an LSTM layer trained on the user's previous interactions:

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

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

$$\begin{aligned} u_{it}^m &= W_m^U z_{it} + B_m^U \\ a_{it}^m &= W_m^A z_{it} + B_m^A, \end{aligned} \tag{3}$$

where W_m^U, W_M^A, B_m^U, and B_m^A 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 PM-F 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:

$$\begin{aligned} u_i^m &= W_m^U z_i + B_m^U \\ a_i^m &= W_m^A z_i + B_m^A. \end{aligned} \tag{4}$$

This keeps the number of model parameters small at a potential cost to model capacity. In contrast, the EM-F 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 item representations are given by the latent vectors e. The input and output item embeddings are tied, and they have the same dimensionality as the user representations.

Experiments

We test our models on a number of publicly available datasets with varying degrees of sparsity as well as diversity of tastes. In all tests, we treat our models as ranking models, and evaluate them on the quality of the ranked recommendation list they generate, rather than the rating predictions they produce (unlike (Wu et al., 2017)).

Datasets

We use the following datasets in our experiments (their summary statistics are listed in Table 1):

1. Movielens 10M: dataset of 10 million movie ratings across 10,000 movies and 72,000 users (Harper and Konstan, 2016).
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 (Zajac, 2017).
3. Amazon: dataset of ratings and reviews gathered from the Amazon online shopping service (Leskovec, Adamic, and Huberman, 2007). 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.

Throughout our experiments, 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.

The two key differences betweeen the datasets are their sparsity and the degree to which they are popularity-biased. The Movielens 10M and Goodbooks datasets are relatively

Table 1: Dataset statistics. $95th/50th$ denotes the ratio of popularity of the 95th and 50th percentile of item popularity.

Dataset	Users	Items	Density	95th/50th
Movielens	69,879	10,678	0.0134	7.42
Amazon	100,085	113,997	0.0003	5.67
Goodbooks	53,425	10,001	0.0112	1.41

dense, while the Amazon dataset is much sparser. Popularity seems to play a greater role in the Movielens dataset than in the other datasets: the ratio between the number of interactions that accrue to the 95th percentile and the 50th percentile of most popular items is highest in Movielens 10M.

We conjecture that taste diversity plays a lesser role in highly popularity-biased datasets (that is, where a large share of all interactions go to very few items). Intuitively, a single dominant taste can model such observations quite well; additional tastes only come into play in the long tail of the distribution. Conversely, a more even distribution of interactions between items allows the possibility that multiple tastes play a larger role. If this is true, we would expect the gains from our mixture models to be larger in the Amazon (where tastes can span multiple unrelated item categories) and Goodbooks (where the popularity distribution is relatively even by construction) datasets than in the Movielens dataset.

Baselines

Our baselines are exact equivalents of the mixture models, differing only in the fact that they represent the user with a single k-dimensional vector. For sequence-based models, we use an LSTM architecture and represent the user directly with the last hidden state of the network (z_{it} from equation 2). For factorization models, we use a standard latent embedding vector, corresponding directly to z_i from the projection mixture model.

This makes our baselines particularly suitable for evaluating mixture-of-tastes representations: adding multiple tastes is the sole architectural difference between the models, and so differences in recommendation performance can be directly attributed to the models' greater ability to model diverse tastes.

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.

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 believe our sequence model experimental setting to be a relatively good reflection of the conditions in which industry recommender system are trained and evaluated: model retraining is often done daily, with data up to the day of training used for model estimation, and subsequent interactions used for evaluation. Insofar as this is true, our results should generalize well to real-world applications.

Loss functions

We experiment with two loss functions:

- Bayesian personalised ranking (BPR, Rendle et al. (2009)), and
- adaptive sampling maximum margin loss, following Weston, Bengio, and Usunier (2011).

For both loss functions, for any known positive user-item interaction pair (i, j), we uniformly sample an implicit negative item k. 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.

Hyperparameter search

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 Melis, Dyer, and Blunsom (2017), 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 algorithms 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.

Implementation

Our models are implemented using the PyTorch deep learning framework (Paszke et al., 2017) and trained using the nVidia K40 GPUs. All of the models are trained using the ADAM (Kingma and Ba, 2014) per-parameter learning rate schedule. We make the model and experiment code (as well as the full results) available on Github[1].

Results

Table 2: Mean reciprocal rank across all users/sequences in the test set. Note that due to differences in experimental protocol, results between sequence-based and factorization models are not directly comparable.

(a) Sequence models

Model	Movielens	Amazon	Goodbooks
LSTM	0.0908	0.1502	0.1158
Mixture-LSTM	**0.1001**	**0.1889**	**0.1358**

(b) Factorization models

Model	Movielens	Amazon	Goodbooks
Bilinear	0.1053	0.1001	0.0738
Projection Mixture	**0.1097**	0.0698	0.0712
Embedding Mixture	0.1026	**0.1696**	**0.0853**

Our main results are summarized in Table 2. In both sequence-based and factorization tasks variants of our model achieve consistently higher performance than baseline models, with improvements ranging from 10% to 69%. The results are robust to optimizing the hyperparameters of our baselines, giving us ample confidence that the results can be replicated in production recommender systems.

Ranking quality

In sequence-based models, our M-LSTM model outperforms the baseline LSTM model on all datasets. The performance gains are particularly large on the Amazon (26%) datset and Goodbooks (17%) datasets, and smaller, but still meaningful, on the Movielens 10M dataset (10%). This is consistent with our conjectures on the nature of the datasets. The Amazon dataset, spanning the entire catalog of Amazon products, benefits most from being able to model taste diversity. This is not surprising, given that a single Amazon user can interact with products from many disparate domains.

Among factorization models, the Embedding Mixture model is the best performing model in two datasets. It beats the baseline model by a very substantial margin on the Amazon dataset (69%), and by a smaller margin of 12% on the Goodbooks dataset. There is relatively little to distinguish the models on the Movielens dataset, where all three models perform equally well. Somewhat surprisngly, the Projection Mixture model fails to outperform the baseline on two datasets, and substantially *underperforms* it on the Amazon dataset. Given that the model is strictly more expressive than the baseline model, we attribute this failure to difficulties in optimization that have eluded us despite experimenting with a number of initialization schemes and learning rate schedules.

Table 3: Effect of number of mixture components

(a) Sequence models

Components	Movielens 10M	Amazon	Goodbooks
2	0.0882	0.1538	0.1262
4	**0.0997**	0.1689	0.1304
6	0.0922	**0.1889**	**0.1358**
8	0.1001	0.1865	0.1327

(b) Factorization models

Components	Movielens 10M	Amazon	Goodbooks
2	0.0931	0.1343	0.0797
4	**0.1026**	0.1583	0.0840
6	0.0864	0.1612	0.0811
8	0.0879	**0.1696**	**0.0853**

Hyperparameter search

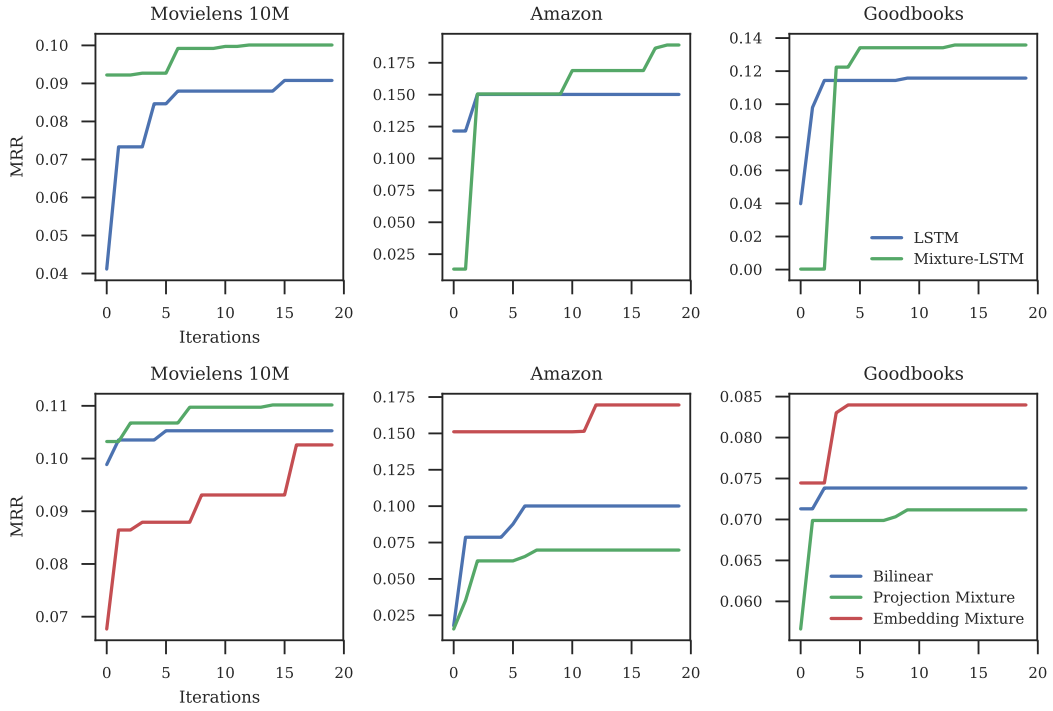
Our results are robust to hyperparameter optimization. Figure 1 plots the maximum test MRR achieved by each algorithm as a function of the number of elapsed hyperparameter search iterations. Both baseline and mixture models benefit from hyperparameter tuning. All algorithms converge to their optimum 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.

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

[1] https://github.com/maciejkula/mixture

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.



recommendation quality: being able to represents more distinct user tasts yields better results. The optimum number of components is dataset dependent: the (more diverse) Amazon and Goodbooks dataset tend to benefit more from additional expressiveness, while four mixture components seem to suffice in the more homogeneous Movielens dataset.

Conclusion

We show that mixture-of-tastes representations are clear improvements over their baseline modes, especially in sequence-based settings where the accuracy gains come at a very modest cost in model complexity. We have taken care to test the mixture models against strong baselines, and have confidence that our approach can translate to accuracy gains in production recommender systems.

References

Athiwaratkun, B., and Wilson, A. G. 2017. Multimodal word distributions. *arXiv preprint arXiv:1704.08424*.

Harper, F. M., and Konstan, J. A. 2016. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 5(4):19.

Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; and Tikk, D. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Hu, Y.; Koren, Y.; and Volinsky, C. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, 263–272. Ieee.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Koren, Y. 2009. The bellkor solution to the netflix grand prize. *Netflix prize documentation* 81:1–10.

Leskovec, J.; Adamic, L. A.; and Huberman, B. A. 2007. The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)* 1(1):5.

Melis, G.; Dyer, C.; and Blunsom, P. 2017. On the state of the art of evaluation in neural language models. *arXiv preprint arXiv:1707.05589*.

Paszke, A.; Gross, S.; Chintala, S.; et al. 2017. Pytorch.

Rendle, S.; Freudenthaler, C.; Gantner, Z.; and Schmidt-Thieme, L. 2009. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, 452–461. AUAI Press.

Vilnis, L., and McCallum, A. 2014. Word representations via gaussian embedding. *arXiv preprint arXiv:1412.6623*.

Weston, J.; Bengio, S.; and Usunier, N. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*, volume 11, 2764–2770.

Weston, J.; Weiss, R. J.; and Yee, H. 2013. Nonlinear latent factorization by embedding multiple user interests. In *Proceedings of the 7th ACM conference on Recommender systems*, 65–68. ACM.

Wu, C.-Y.; Ahmed, A.; Beutel, A.; Smola, A. J.; and Jing, H. 2017. Recurrent recommender networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 495–503. ACM.

Zajac, Z. 2017. Goodbooks-10k dataset. *GitHub*.