TAFA: Two-headed Attention Fused Autoencoder for Context-Aware Recommendations

Jin Peng Zhou*
University of Toronto, Layer 6 AI
jinpeng.zhou@mail.utoronto.ca

Zhaoyue Cheng*
Layer 6 AI
joey@layer6.ai

Felipe Pérez
Layer 6 AI
felipe@layer6.ai

Maksims Volkovs
Layer 6 AI
maks@layer6.ai

ABSTRACT

Collaborative filtering with implicit feedback is a ubiquitous class of recommendation problems where only positive interactions such as purchases or clicks are observed. Autoencoder-based recommendation models have shown strong performance on many implicit feedback benchmarks. However, these models tend to suffer from popularity bias making recommendations less personalized. User-generated reviews contain a rich source of preference information, often with specific details that are important to each user, and can help mitigate the popularity bias. Since not all reviews are equally useful, existing work has been exploring various forms of attention to distill relevant information. In the majority of proposed approaches, representations from implicit feedback and review branches are simply concatenated at the end to generate predictions. This can prevent the model from learning deeper correlations between the two modalities and affect prediction accuracy. To address these problems, we propose a novel Two-headed Attention Fused Autoencoder (TAFA) model that jointly learns representations from user reviews and implicit feedback to make recommendations. We apply early and late modality fusion which allows the model to fully correlate and extract relevant information from both input sources. To further combat popularity bias, we leverage the Noise Contrastive Estimation (NCE) objective to "de-popularize" the fused user representation via a two-headed decoder architecture. Empirically, we show that TAFA outperforms leading baselines on multiple real-world benchmarks. Moreover, by tracing attention weights back to reviews we can provide explanations for the generated recommendations and gain further insights into user preferences. Full code for this work is available here: https://github.com/layer6ai-labs/TAFA.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; **Collaborative filtering**; • **Computing methodologies** → **Neural networks**.

*Both authors contributed equally to the paper

RecSys '20, September 22–26, 2020, Virtual Event, Brazil

ACM ISBN 978-1-4503-7583-2/20/09.
https://doi.org/10.1145/3383313.3412268

KEYWORDS

Deep Learning, Context-Aware Recommender Systems, Neural Attention Networks

ACM Reference Format:
Jin Peng Zhou, Zhaoyue Cheng, Felipe Pérez, and Maksims Volkovs. 2020. TAFA: Two-headed Attention Fused Autoencoder for Context-Aware Recommendations. In *Fourteenth ACM Conference on Recommender Systems (RecSys '20), September 22–26, 2020, Virtual Event, Brazil.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3383313.3412268

1 INTRODUCTION

With virtually unlimited product choices, high-quality recommender systems have become an essential part of online platforms, and serve to help customers find items of interest in every aspect of their daily lives. Collaborative filtering (CF) is the de-facto approach for making personalized recommendations based on user-item interactions [18]. In many cases interactions such as purchase history are taken as positive signals while explicit negative signals are missing. Absence of interaction does not necessarily imply negative preference as user can be unaware of the particular item. However, in many applications only positive interactions are available, and the focus of recent research has been on modeling such data [15, 24], we refer to this as the implicit feedback setting.

A diverse set of approaches have been proposed to tackle recommendations in the implicit feedback setting. For example, WRMF [10] characterizes users and items with latent factors learned via matrix factorization; while metric learning approach CML [8], learns a joint user-item metric to model similarity. Notably, autoencoder-based methods have shown to be effective in this setting [12, 21, 26]. Autoencoders learn preference patterns by first encoding observed interactions into a lower-dimensional latent representation. This representation is then decoded to reconstruct observed data and simultaneously generate predictions for unobserved interactions. By using non-linear activation functions in the encoding layer, an autoencoder is able to represent more complex relationships between users and items than linear embedding approaches such as matrix factorization. However, a vanilla autoencoder often exhibits bias towards popular items, making recommendations less personalized and limiting accuracy [25].

User-generated text reviews can be an important source of preference data, and often contain very specific and nuanced information about user preferences. High-quality reviews typically have particular reasons why a user likes or dislikes a particular item. Leveraging this additional information can be an effective way to mitigate the

popularity bias, and improve the overall performance of the recommendation model. One major challenge with utilizing reviews lies in the fact that the quality of reviews vary. Not all reviews are equally informative, and not all words/sections within a particular review contain useful information. One way to address this challenge is to use attention over reviews or concepts extracted from them. This can allow the model to focus on parts that are helpful for recommendation and discount everything else. Although previous works that utilize various attention mechanisms have achieved considerable improvement in performance [4, 13, 14], one major bottleneck is the lack of an effective way to correlate meaningful information between observed preferences and reviews. Majority of existing approaches simply concatenate both representations to make predictions. This prevents deeper interactions between the two modalities and can affects recommendation performance.

To address these issues, we propose a novel approach called **T**wo-headed **A**ttention **F**used **A**utoencoder (TAFA). TAFA consists of several components: a preference encoder which encodes implicit feedback, a review encoder which applies attention mechanism at both word and review levels, early and late fusion modules to combine information from both encoders, and a decoder with two decoder heads. To better correlate the representations from the two modalities, we propose to first apply an early fusion module which leverages attention to select which reviews are relevant for the recommendation task. This is followed by a late fusion module which combines representations from the two encoders. Finally, we use a two-headed decoder architecture to further mitigate popularity bias through a closed-form Noise Contrastive Estimation [25, 29]. Extensive empirical evaluation on six public benchmarks shows that TAFA outperforms many leading baselines. We further demonstrate that our approach leads to a higher level of personalization by accurately recommending less popular items to users. Finally, by tracing attention weights back to the reviews we can explain generated recommendations and gain further insight into user preferences.

2 RELATED WORK

Deep Learning Given the success of deep learning approaches in fields such as computer vision and natural language processing, a number of deep learning approaches have been proposed for the implicit feedback recommendation problem [7, 8, 12, 20, 26]. Even though classical techniques such as neighbor similarity remain strong baselines [5], deep learning approaches have shown promising performance particularly when other sources of data such as images or text are available [27]. Of these, autoencoder-based models have shown strong performance on a number on benchmarks, with an ability to learn robust representations for users and items. One of the earliest such approaches is AutoRec [20], an encoder-decoder architecture that maps users' implicit feedback to a low-dimensional latent space via a feed forward network. A symmetric architecture is then used to decode, and the model is optimised with a reconstruction loss. Following the success of AutoRec, other related approaches have been proposed including Collaborative Denoising Autoencoder [26] and Variational Autoencoder [12]. One prominent challenge in recommender systems is bias towards popular items [1]. As we demonstrate empirically this bias can be particularly strong in autoencoder models limiting their accuracy, similar observation was also made by [25].

User Reviews In many applications, other sources of information are often available in addition to implicit feedback. One important such source is user-generated textual reviews. Reviews can contain rich preference information which can be leveraged to augment implicit feedback. To model user reviews many recent works, motivated by the advances in natural language processing, have proposed to use deep learning methods that jointly learn from reviews and implicit feedback [3, 4, 9, 13, 22, 28]. Early work in this area includes DeepCoNN [28] and TransNet [3]. Both approaches use convolutional neural networks (CNNs) on review word embeddings to generate review-based user representations. These representations are then used in a factorization machine to make predictions. To model multiple reviews from the same user, DeepCoNN and TransNet concatenate reviews together. This forces the CNN to model all reviews which can skew representations since not every review contains useful information. Later approaches address this by introducing attention [4, 13, 14], which enables the model to focus on relevant information. Specifically, [4] applies review-level attention on the text representations learned by the CNN. On the other hand, [13] first extracts topics from reviews and then applies attention over topics in a recurrent model. Finally, [14] applies a word-level attention mechanism to encode text followed by a gating layer analogous to the LSTM, output of this layer is then fed to an autoencoder to make predictions. However, one major drawback of all of these approaches is that the review and implicit feedback representations are simply concatenated together, limiting model's ability to learn deeper interactions between the two modalities.

Noise Contrastive Estimation Recommender systems often exhibit a strong popularity bias that prevents them from accurately recommending less popular items [1, 2]. To reduce this bias, Noise Contrastive Estimation (NCE) [6] aims to discriminate between the observed data and artificially generated noise that usually comes from a popularity-based distribution. Furthermore, in the implicit feedback setting, NCE becomes a robust tool to learn a recommendation model without explicitly assuming that missing interactions indicate negative preference as done in other models [25]. In this setting, the NCE objective aims to increase the likelihood of observed implicit interactions, while minimizing the probability of negative samples drawn from a popularity-based noise distribution.

3 APPROACH

Given users $\{u\}_{u \in U}$ and items $\{i\}_{i \in I}$, the implicit feedback r_{ui} for user u on item i is set to be 1 if there is a positive historical interaction between them. If there is no such interaction we set $r_{ui} = 0$. The $|U| \times |I|$ matrix R obtained from r_{ui} across all users and items is referred to as the *implicit feedback matrix*. We note that the column $r_{:,i}$ represents the historical feedback for item i across all users. Similarly, $r_{u,:}$ contains the complete historical information for user u. In this work we propose an end-to-end model, TAFA, that leverages both implicit feedback and review information. Given the implicit feedback matrix R and a set of reviews, we aim to first encode R into a lower dimension and then reconstruct it filling missing values in the process. We denote this reconstruction by $\tilde{R}$. TAFA contains the following major components: preference

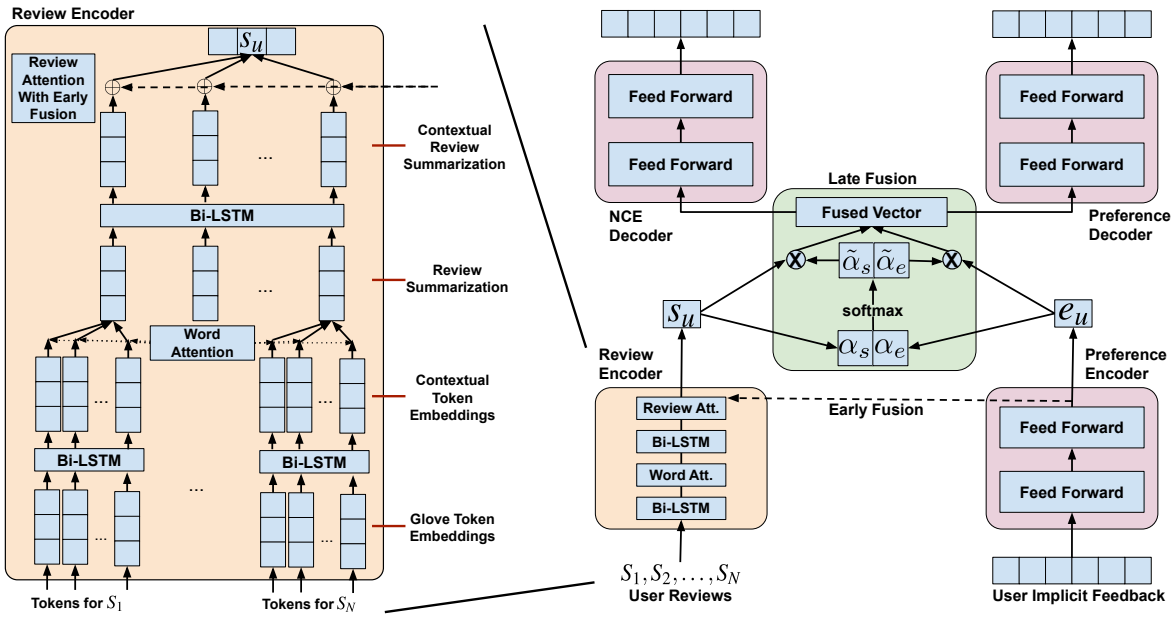


Figure 1: TAFA model architecture. Review encoder encodes user reviews into user review representation. Preference encoder encodes user implicit feedback into user preference representation. Early and late fusion modules integrate the two representations together. NCE and preference decoders reconstruct by minimising the NCE and squared error losses respectively.

encoder which encodes historical user-item interactions, review encoder which extracts relevant information from the reviews, two fusion modules that integrate the representations from both encoders and a two-headed decoder. The full model architecture is shown in Figure 1, and we describe each component in detail below.

Preference Encoder The preference encoder uses implicit feedback to generate a latent representation for each user. Given a user u with implicit feedback $r_{u,:}$, we encode $r_{u,:}$ via a two-layer feed-forward MLP to obtain latent representation e_u. This architecture is analogous to [20], and the main difference is that we apply noise to the input in the form of dropout [26]. Adding input noise makes this module similar to the denoising autoencoder [23] and improves generalisation. It is worth noting here that the latent representations learned by this model geometrically encode information about the number of interactions that each user has, see Section 4.8 for more details.

Review Encoder User-generated reviews contain important preference information with specific details about items. To extract this information an important step is to distinguish between relevant and noisy reviews. Furthermore, within each review it is desirable to identify particularly important sections. Guided by this intuition, we use attention combined with recurrent neural networks to extract relevant information at both word and review level. Let $S_1, \ldots, S_N$ denote a sequence of reviews written by a given user. Each review S is composed of word tokens $t_1, \ldots, t_{|S|}$. We first embed these tokens using the pre-trained GloVe embeddings [16]. To further capture contextual information, an embedded token sequence is passed through a bi-directional LSTM. Hidden states in both directions at each token are then concatenated together to produce contextual token embeddings $\hat{t}_1, \ldots, \hat{t}_{|S|}$. After contextual encoding, we apply attention which allows the model to focus on relevant tokens within each review:

$$\begin{aligned} \gamma_k &= W_2 \tanh(W_1 \hat{t}_k + b_1) + b_2 \\ a_k &= \frac{\exp(\gamma_k)}{\sum_{k'=1}^{|S|} \exp(\gamma_{k'})} \\ a &= \sum_{k=1}^{|S|} a_k \cdot \hat{t}_k \end{aligned} \tag{1}$$

where W's and b's are weights and biases to be learned. The vector a can be interpreted as a summarization for the review S. Repeating this process for every user review $S_1, \ldots, S_N$, we obtain the corresponding attention vectors $a_1, \ldots, a_N$. Similarly to contextualising words, we apply another bi-directional LSTM over the attention vectors, and concatenate hidden states in both directions at each review to get attended contextualised review vectors $\hat{a}_1, \ldots, \hat{a}_N$. These vectors capture both global context across reviews and specific word-level information from each review. In practice users can have hundreds or even thousands of reviews. To make this computation practical we can sample a subset of the most recent reviews as those are more likely to convey the latest user preference.

Early Fusion The objective of the early fusion is to combine implicit feedback with the information extracted from reviews before it is aggregated to form a review-based user representation. To this end we fuse the contextualised review vectors $\hat{a}_1, \ldots, \hat{a}_N$ with attention that incorporates the user representation e_u from the preference encoder. By doing so, the selection of important reviews relies not only on review information, but also on the implicit feedback from the user making it more accurate. To fuse the information from the two sources, we concatenate user representation e_u from

Table 1: Dataset statistics.

Dataset	# Users	# Items	# Ratings	# Review Words	Density
Yelp 2013	1,631	1,633	78,966	6,844,429	2.96×10^{-2}
Yelp 2014	4,818	4,194	231,163	18,324,710	1.14×10^{-2}
Amazon Digital Music	5,541	3,568	64,706	2,561,491	3.27×10^{-3}
Amazon Grocery and Gourmet Food	14,681	8,713	151,254	3,074,602	1.18×10^{-3}
Amazon Video Games	24,383	18,672	231,780	8,385,355	5.09×10^{-4}
Amazon CDs and Vinyl	75,258	64,443	1,097,592	29,254,672	2.26×10^{-4}

the preference encoder with each review vector before applying attention. Using attention weights, we then combine all reviews to get user review representation s_u:

$$\begin{aligned} \beta_n &= W_4 \tanh(W_3[\hat{a}_n; e_u] + b_3) + b_4 \\ g_n &= \frac{\exp(\beta_n)}{\sum_{n'=1}^{N} \exp(\beta_{n'})} \\ s_u &= \sum_{n=1}^{N} g_n \cdot \hat{a}_n \end{aligned} \tag{2}$$

where $[x; y]$ is the concatenation operation. Note that without early fusion, once review vectors are aggregated together the information from individual reviews is lost. So fusing at a later stage, as done in other models [4, 13], has a disadvantage where the model has less flexibility to focus on information from specific reviews.

Late Fusion Using the two encoders we obtain a preference representation e_u and a review representation s_u for each user. The latent spaces for preference and review encoders can differ and their contribution to the final prediction may vary, so simply concatenating the two representations together can be inadequate. To ensure that the information from two representations is properly combined we introduce late stage fusion. We use cross modal attention and first map each representation to a common latent space. Attention is then applied in this space to fuse the two representations:

$$\begin{aligned} \alpha_s &= W_5 \tanh(W_6 s_u + b_6) + b_5 \\ \alpha_e &= W_5 \tanh(W_7 e_u + b_7) + b_5 \\ \tilde{\alpha_s}, \tilde{\alpha_e} &= \text{softmax}(\alpha_s, \alpha_e) \\ v_s &= W_v \tanh(W_6 s_u + b_6) + b_v \\ v_e &= W_v \tanh(W_7 e_u + b_7) + b_v \\ v_{\text{fused}} &= \tilde{\alpha_s} \cdot v_s + \tilde{\alpha_e} \cdot v_e \end{aligned} \tag{3}$$

where v_{fused} is the final user representation that combines both modalities. By sharing weights W_v and biases b_v we ensure that the two representations are mapped to a common space before fusion. Similarly, by sharing attention weights W_5 and b_5 we also ensure that attention coefficients are mapped to the same space before the softmax normalisation. Fusing with softmax attention has a clear advantage over concatenation in that the model can explicitly decide how much weight is given to each representation.

Two-Headed Decoder As we discussed, autoencoder models tend to exhibit excessive bias towards popular items. To address this problem we leverage the NCE framework. The NCE objective aims to increase the likelihood of observed interactions, while minimizing it for negative samples drawn from a popularity-based noise distribution q [6]:

$$\underset{\theta}{\text{argmin}} - \sum_{i} r_{u,i} \left[\log p(r_{u,i} = 1) + E_{q(i')}[\log p(r_{u,i'} = 0)] \right] \tag{4}$$

where θ is the set of free-parameters to be learned. Both probabilities are modelled using the decoder:

$$p(r_{u,i} = 1) = \sigma(\tilde{r}_{u,i}; \theta) \quad \textit{and} \quad p(r_{u,i'} = 0) = 1 - \sigma(\tilde{r}_{u,i'}; \theta) \tag{5}$$

where $\tilde{r}_{u,i}$ is the reconstructed interaction, and σ is the sigmoid function. Combining Equations 4 and 5 we can solve for the reconstructed matrix $\tilde{R}$ analytically [25, 29] by noting that:

$$\frac{\partial \ell}{\partial \tilde{r}_{u,i}} = \sigma(-\tilde{r}_{u,i}) - \frac{|r_{:,i}|}{\sum_{i'} |r_{:,i'}|} \sigma(\tilde{r}_{u,i}) \tag{6}$$

where ℓ is the loss in Equation 4. As a result, the optimal solution $r^*_{u,i}$ for the observed interaction is:

$$r^*_{u,i} = \log \frac{\sum_{i'} |r_{:,i'}|}{|r_{:,i}|} \qquad \forall r_{u,i} = 1 \tag{7}$$

and for the unobserved interactions the optimal solution is simply:

$$r^*_{u,i} = 0 \qquad \forall r_{u,i} = 0 \tag{8}$$

We build on this framework and use a two-headed decoder that is optimized for both reconstruction and NCE losses simultaneously. We hypothesize that the analytic NCE solution in Equation 7 can be used as an effective way to "de-popularize" the representations learned by the encoders, and can also serve as a form of regularisation. Analogous to reconstruction loss, we also use the squared error objective in the NCE head but with $r^*_{u,i}$ as the target. Empirically we demonstrate that adding this loss does decrease the popularity bias making recommendations more personalized and improving accuracy (see Section 4.7).

As seen in Figure 1, each decoder head takes as input the fused representation v_{fused}, transforms it with a two-layer MLP and passes it to the target loss for each head. The NCE head aims to minimize the difference between decoder reconstruction and the analytic NCE solution from Equation 7:

$$L^{NCE}_u = \left\| r^*_{u,:} - h_{\text{nce}}(v_{\text{fused}}) \right\|_2 \tag{9}$$

where h_{nce} is the NCE decoder. Similarly, the reconstruction head is optimized with the mean squared error (MSE) reconstruction objective:

$$L^{MSE}_u = \left\| r_{u,:} - h_{\text{mse}}(v_{\text{fused}}) \right\|_2 \tag{10}$$

where h_{mse} is the MSE decoder. During training we linearly combine the two losses:

$$L = \sum_{u} L^{MSE}_u + L^{NCE}_u + \lambda \left\| \theta \right\|^2 \tag{11}$$

Table 2: Model performance on the Yelp 2013 dataset.

	model	R-Precision	NDCG	Precision@5	Precision@20	Precision@50	Recall@5	Recall@20	Recall@50
Baselines	AutoRec	0.0187±0.0017	0.0632±0.0019	0.0191±0.0016	0.0134±0.0008	0.0100±0.0005	0.0296±0.0006	0.0842±0.0010	0.1498±0.0014
	CDAE	0.0183±0.0017	0.0637±0.0019	0.0194±0.0016	0.0136±0.0009	0.0101±0.0005	0.0300±0.0006	0.0843±0.0009	0.1538±0.0013
	CML	0.0128±0.0017	0.0411±0.0018	0.0139±0.0016	0.0089±0.0009	0.0064±0.0005	0.0214±0.0005	0.0519±0.0009	0.0927±0.0012
	VAE-CF	0.0265±0.0017	0.0780±0.0019	0.0253±0.0016	0.0164±0.0009	0.0117±0.0005	0.0393±0.0005	0.1016±0.0010	0.1769±0.0014
	WRMF	0.0210±0.0017	0.0653±0.0018	0.0202±0.0016	0.0133±0.0008	0.0101±0.0005	0.0304±0.0005	0.0827±0.0010	0.1524±0.0014
	TARMF	0.0270±0.0017	0.0868±0.0019	0.0261±0.0016	0.0180±0.0009	0.0127±0.0004	0.0420±0.0005	0.1135±0.0010	0.1971±0.0014
	GATE	0.0223±0.0017	0.0894±0.0019	0.0236±0.0016	0.0178±0.0009	0.0140±0.0004	0.0404±0.0005	0.1146±0.0010	0.2155±0.0014
Our Model	TAFA	**0.0343±0.0017**	**0.0995±0.0018**	**0.0304±0.0016**	**0.0194±0.0009**	**0.0144±0.0005**	**0.0495±0.0007**	**0.1223±0.0009**	**0.2232±0.0014**

Table 3: Model performance on the Yelp 2014 dataset.

	model	R-Precision	NDCG	Precision@5	Precision@20	Precision@50	Recall@5	Recall@20	Recall@50
Baselines	AutoRec	0.0187±0.0008	0.0648±0.0017	0.0182±0.0006	0.0129±0.0004	0.0095±0.0002	0.0308±0.0003	0.0859±0.0006	0.1551±0.0008
	CDAE	0.0166±0.0007	0.0581±0.0017	0.0171±0.0005	0.0115±0.0004	0.0086±0.0004	0.0286±0.0003	0.0775±0.0006	0.1390±0.0007
	CML	0.0147±0.0007	0.0483±0.0017	0.0148±0.0005	0.0099±0.0004	0.0073±0.0004	0.0244±0.0003	0.0628±0.0006	0.1127±0.0007
	VAE-CF	0.0170±0.0008	0.0604±0.0017	0.0173±0.0006	0.0120±0.0004	0.0090±0.0004	0.0285±0.0003	0.0794±0.0006	0.1432±0.0007
	WRMF	0.0267±0.0008	0.0861±0.0017	0.0258±0.0005	0.0173±0.0005	0.0120±0.0004	0.0452±0.0002	0.1185±0.0006	0.1993±0.0008
	TARMF	0.0272±0.0008	0.0884±0.0017	0.0263±0.0005	0.0176±0.0006	0.0125±0.0004	0.0451±0.0004	0.1170±0.0006	0.2060±0.0008
	GATE	0.0221±0.0008	0.0811±0.0016	0.0216±0.0005	0.0158±0.0006	0.0119±0.0002	0.0381±0.0004	0.1084±0.0006	0.1967±0.0008
Our Model	TAFA	**0.0319±0.0008**	**0.0990±0.0017**	**0.0307±0.0005**	**0.0196±0.0002**	**0.0138±0.0004**	**0.0528±0.0003**	**0.1298±0.0006**	**0.2266±0.0008**

Table 4: Model performance on the Amazon Digital Music dataset.

	model	R-Precision	NDCG	Precision@5	Precision@20	Precision@50	Recall@5	Recall@20	Recall@50
Baselines	AutoRec	0.0152±0.0020	0.0612±0.0021	0.0138±0.0016	0.0087±0.0008	0.0062±0.0004	0.0376±0.0005	0.0938±0.0020	0.1658±0.0025
	CDAE	0.0063±0.0016	0.0372±0.0021	0.0051±0.0010	0.0048±0.0007	0.0039±0.0004	0.0159±0.0005	0.0583±0.0020	0.1137±0.0023
	CML	0.0117±0.0020	0.0379±0.0023	0.0085±0.0010	0.0050±0.0008	0.0035±0.0004	0.0259±0.0005	0.0570±0.0021	0.0950±0.0025
	VAE-CF	0.0374±0.0020	0.1074±0.0023	0.0274±0.0016	0.0149±0.0008	0.0092±0.0004	0.0806±0.0005	0.1640±0.0020	0.2450±0.0025
	WRMF	0.0357±0.0020	0.1150±0.0022	0.0264±0.0016	0.0153±0.0007	0.0100±0.0004	0.0805±0.0005	0.1765±0.0019	0.2820±0.0023
	TARMF	0.0393±0.0018	0.1210±0.0022	0.0314±0.0016	0.0163±0.0008	0.0102±0.0004	0.0944±0.0005	0.1870±0.0020	0.2855±0.0025
	GATE	0.0543±0.0020	0.1557±0.0023	0.0395±0.0016	0.0205±0.0008	0.0126±0.0004	0.1227±0.0004	0.2418±0.0019	0.3572±0.0025
Our Model	TAFA	**0.0646±0.0020**	**0.1723±0.0022**	**0.0441±0.0016**	**0.0227±0.0008**	**0.0130±0.0004**	**0.1366±0.0005**	**0.2674±0.0020**	**0.3767±0.0025**

where θ is the full set of parameters to be learned and λ is weight penalty. The gradients from each head are combined and back-propagated through the entire architecture to review token embeddings that are also updated during training. At inference, we use the reconstruction from the MSE decoder head to make predictions. We find that this gives better accuracy than combining the predictions from the two decoders as the NCE head is used primarily for de-biasing and regularisation.

4 EXPERIMENTS

We evaluate the performance of our approach on six real-wold datasets, and compare to both classic collaborative filtering methods and leading recent deep learning approaches. We conduct an extensive ablation study to investigate how different hyper-parameters and modules affect recommendation accuracy in our model. To evaluate the effect of the NCE objective we compute popularity bias for all models and analyse the level of personalisation. Finally, we conduct a qualitative investigation by tracing attention weights back to the review text and visualising parts that the model is focusing on.

4.1 Datasets and Evaluation Metrics

We use two Yelp[1] and four Amazon[2] datasets to benchmark model performance. All datasets have ratings and user reviews and vary in size, domain and sparsity, giving a broad view on the applicability of our approach. The summary statistics of each dataset are shown in Table 1. For each of the six datasets, we first randomly split it into training, validation and test sets with 80%, 10% and 10% splits. To simulate implicit feedback we follow existing literature [14, 19], and set ratings greater than the threshold (three) to 1 and those less than or equal to the threshold to 0. We tune hyper-parameters of all methods on the validation set, and evaluate final performance on the test set by calculating four standard evaluation metrics: R-Precision [17], NDCG [17], Precision@K and Recall@K with $K \in \{5, 20, 50\}$. All results are reported with a 95% confidence interval. R-precision is order insensitive and uses the number of interaction for each user as relevance cut-off. NDCG is order sensitive and discounts each recommendation by its rank. Precision@K/Recall@K are similar to R-precision but use a fixed cut-off K for each user. Together these metrics evaluate different aspects of the recommendation task and provide a thorough analysis of model performance.

4.2 Baseline Models

Recommender systems has been a popular research area and many approaches have been proposed. Despite significant progress, factorisation methods still provide highly competitive performance [5], and we use Weighted Regularized Matrix Factorization (WRMF) [10] and Collaborative Metric Learning (CML) [8] as our benchmarks from this category. Autoencoder models have also shown highly

[1] https://www.yelp.com/dataset
[2] http://jmcauley.ucsd.edu/data/amazon

Table 5: Model performance on the Amazon Grocery and Gourmet Food dataset.

	model	R-Precision	NDCG	Precision@5	Precision@20	Precision@50	Recall@5	Recall@20	Recall@50
Baselines	AutoRec	0.0079±0.0007	0.0328±0.0010	0.0059±0.0005	0.0041±0.0004	0.0033±0.0003	0.0149±0.0005	0.0462±0.0012	0.0980±0.0019
	CDAE	0.0078±0.0006	0.0329±0.0010	0.0056±0.0005	0.0042±0.0004	0.0033±0.0003	0.0141±0.0005	0.0476±0.0014	0.0977±0.0018
	CML	0.0109±0.0007	0.0415±0.0011	0.0089±0.0004	0.0054±0.0004	0.0039±0.0003	0.0259±0.0005	0.0631±0.0013	0.1120±0.0019
	VAE-CF	0.0125±0.0007	0.0534±0.0011	0.0112±0.0005	0.0074±0.0004	0.0050±0.0003	0.0333±0.0005	0.0863±0.0014	0.1442±0.0017
	WRMF	0.0219±0.0007	0.0598±0.0011	0.0144±0.0005	0.0076±0.0004	0.0046±0.0003	0.0454±0.0005	0.0906±0.0014	0.1364±0.0019
	TARMF	0.0162±0.0007	0.0596±0.0011	0.0124±0.0004	0.0082±0.0004	0.0055±0.0003	0.0373±0.0005	0.0958±0.0014	0.1567±0.0018
	GATE	0.0173±0.0007	0.0716±0.0011	0.0153±0.0005	0.0096±0.0004	**0.0064±0.0003**	0.0479±0.0005	0.1188±0.0014	**0.1916±0.0019**
Our Model	TAFA	**0.0239±0.0007**	**0.0762±0.0011**	**0.0166±0.0005**	**0.0102±0.0004**	0.0063±0.0003	**0.0508±0.0005**	**0.1221±0.0014**	0.1900±0.0019

Table 6: Model performance on the Amazon Video Games dataset.

	model	R-Precision	NDCG	Precision@5	Precision@20	Precision@50	Recall@5	Recall@20	Recall@50
Baselines	AutoRec	0.0061±0.0002	0.0265±0.0003	0.0048±0.0003	0.0031±0.0002	0.0024±0.0001	0.0154±0.0001	0.0401±0.0002	0.0777±0.0002
	CDAE	0.0061±0.0002	0.0270±0.0002	0.0049±0.0003	0.0032±0.0002	0.0024±0.0001	0.0163±0.0001	0.0424±0.0001	0.0788±0.0002
	CML	0.0125±0.0002	0.0392±0.0003	0.0070±0.0003	0.0041±0.0002	0.0033±0.0002	0.0228±0.0002	0.0523±0.0002	0.1062±0.0002
	VAE-CF	0.0214±0.0002	0.0709±0.0003	0.0156±0.0002	0.0088±0.0002	0.0056±0.0001	0.0506±0.0002	0.1130±0.0002	0.1794±0.0002
	WRMF	0.0194±0.0002	0.0695±0.0003	0.0150±0.0002	0.0085±0.0002	0.0056±0.0001	0.0492±0.0002	0.1098±0.0002	0.1787±0.0002
	TARMF	0.0205±0.0002	0.0655±0.0003	0.0141±0.0003	0.0079±0.0003	0.0052±0.0001	0.0462±0.0002	0.1024±0.0002	0.1637±0.0002
	GATE	0.0216±0.0002	0.0738±0.0003	0.0152±0.0004	0.0089±0.0002	0.0058±0.0001	0.0513±0.0002	0.1172±0.0002	0.1887±0.0002
Our Model	TAFA	**0.0288±0.0003**	**0.0897±0.0003**	**0.0201±0.0004**	**0.0109±0.0002**	**0.0068±0.0001**	**0.0665±0.0001**	**0.1435±0.0002**	**0.2178±0.0002**

Table 7: Model performance on the Amazon CDs and Vinyl dataset.

	model	R-Precision	NDCG	Precision@5	Precision@20	Precision@50	Recall@5	Recall@20	Recall@50
Baselines	AutoRec	0.0018±0.0001	0.0079±0.0002	0.0016±0.0003	0.0013±0.0002	0.0010±0.0001	0.0033±0.0001	0.0117±0.0002	0.0231±0.0002
	CDAE	0.0017±0.0001	0.0080±0.0001	0.0015±0.0003	0.0012±0.0002	0.0010±0.0001	0.0032±0.0001	0.0116±0.0002	0.0233±0.0002
	CML	0.0161±0.0001	0.0490±0.0002	0.0128±0.0003	0.0075±0.0002	0.0049±0.0001	0.0331±0.0001	0.0732±0.0001	0.1169±0.0002
	VAE-CF	0.0187±0.0001	0.0522±0.0002	0.0139±0.0003	0.0076±0.0002	0.0047±0.0001	0.0385±0.0001	0.0791±0.0002	0.1188±0.0002
	WRMF	0.0135±0.0001	0.0434±0.0002	0.0113±0.0003	0.0069±0.0002	0.0046±0.0001	0.0278±0.0001	0.0661±0.0002	0.1067±0.0003
	TARMF	0.0175±0.0001	0.0503±0.0002	0.0138±0.0003	0.0077±0.0002	0.0049±0.0001	0.0353±0.0001	0.0751±0.0002	0.1165±0.0002
	GATE	0.0216±0.0001	0.0586±0.0002	0.0165±0.0003	0.0091±0.0002	0.0056±0.0001	0.0439±0.0001	0.0894±0.0002	0.1284±0.0002
Our Model	TAFA	**0.0224±0.0001**	**0.0662±0.0002**	**0.0171±0.0003**	**0.0098±0.0002**	**0.0061±0.0001**	**0.0456±0.0001**	**0.1012±0.0001**	**0.1559±0.0002**

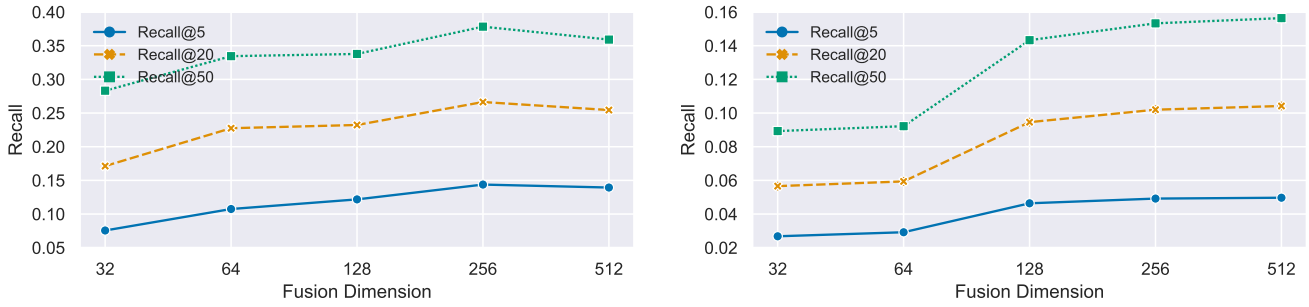
competitive performance on implicit feedback. Among the many proposed models, the most relevant to our work are the Collaborative Denoising Autoencoder (CDAE) [26], Variational Autoencoder for Collaborative Filtering (VAE-CF) [12] and AutoRec [20]. Since our model incorporates text review information we also benchmark Topical Attention Regularized Matrix Factorization (TARMF) [13], a state-of-the-art matrix factorization model that uses textual feature attention to learn user and item representations. Finally, we compare against the Gatted Attentive Autoencoder (GATE) [14], as it is the model closest to our work which incorporates both autoencoder and review information. In total, we compare to seven baselines that cover a wide spectrum of proposed methods in this area.

4.3 Hyper-parameter Settings

We tune all the hyper-parameters on the validation set via grid search. To ensure fair comparison all important parameters such as embedding dimensions and regularisation are swept over the same grid for each model. For all models we search the dimension for user and item embeddings in $\{50, 100, 200, 500\}$, and L^2 regularization weight penalty in $\{1e^{-7}, 1e^{-6}, \ldots, 1e^{4}\}$. For WRMF we select the loss weighting in $\{-0.5, -0.4, \ldots, -0.1\} \cup \{0, 0.1, 1, 10, 100\}$. The corruption parameter for CDAE and VAE-CF is chosen in $\{0.1, 0.2, \ldots, 0.5\}$, and the confidence matrix coefficient for GATE is chosen in $\{5, 10, 15, 20\}$. The hidden dimension for review encoder in TARMF and our model is selected from $\{32, 64, 128, 256\}$. Finally, we choose the fusion dimension for TAFA in $\{32, 64, 128, 256, 512\}$ and discuss the effect of this parameter below. All training and parameter selection is done on a workstation with 40 Intel Xeon 2.20GHz CPUs, 256GB RAM and Titan V GPU.

4.4 Performance Comparison

The performance of all methods on each of the six datasets is shown in Tables 2-7. From the tables we see that TAFA outperforms all baselines on all datasets and metrics except Amazon Grocery and Gourmet Food. On the Amazon Grocery and Gourmet Food dataset GATE performs better on Precision and Recall at higher truncation $K = 50$. These results demonstrate that TAFA has highly robust and stable performance beating leading baselines by a wide margin in many cases. We also observe that the performance of TARMF and GATE are generally better than other baselines including VAE-CF and WRMF. This indicates that incorporating text reviews can benefit the recommendation task significantly due to the extra preference information contained in them. Performance of TAFA is generally strongest on the two largest datasets Amazon Video Games and Amazon CDs. Fusion and encoder-decoder modules together with word embeddings and contextual LSTMs have considerable number of free parameters. Strong performance on larger datasets suggests that our model performs best when there is sufficient data to properly train all components. We anticipate a further performance gain on larger datasets and leave this investigation for future work.



(a) Amazon Digital Music dataset. **(b) Amazon CDs and Vinyl dataset.**

Figure 2: Recall plots for TAFA on two Amazon datasets as dimension of v_{fused} is varied from 32 to 512. Error bars are too small to be shown.

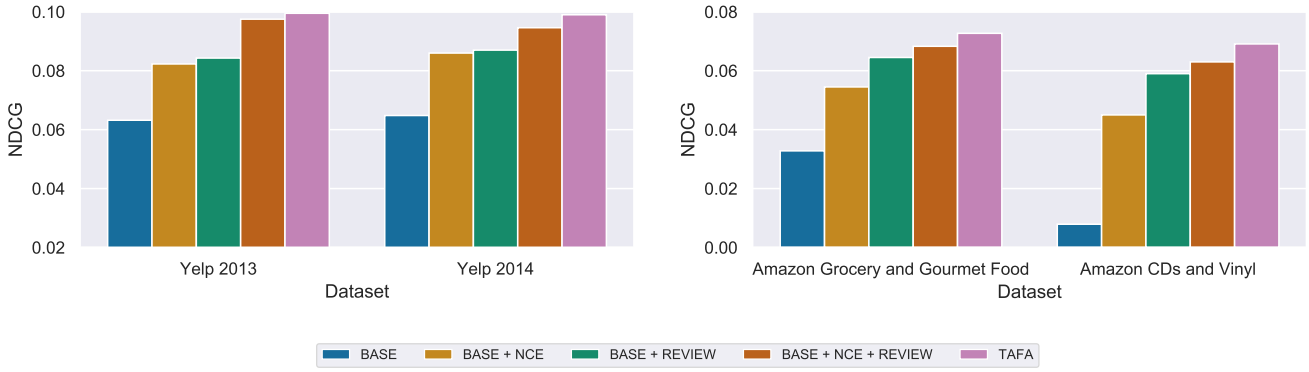


Figure 3: TAFA ablation NDCG results on four datasets. BASE mode has preference encoder and preference (MSE) decoder only. BASE + NCE adds NCE decoder head, BASE + REVIEW adds review encoder, and BASE + NCE + REVIEW has both NCE decoder and review encoder. Both BASE + REVIEW and BASE + NCE + REVIEW use concatenation instead of fusion to combine representations from the two encoders. BASE + NCE + REVIEW thus differs from TAFA in that TAFA applies early and late stage fusion to combine the two modalities. Results on other datasets and metrics show similar trends.

4.5 Fusion Dimension

The dimension of v_{fused} (see Equation 3) can be viewed as an information bottleneck after merging representations from implicit feedback and user reviews. Both decoders receive v_{fused} as input so its size determines the amount of information that is available for decoding and consequently for prediction. To understand how the size of this bottleneck affects performance we plot Recall@5, Recall@20 and Recall@50 as the fusion dimension is varied in {32, 64, 128, 256, 512}. Results on the Amazon Digital Music and Amazon CDs and Vinyl datasets are shown in Figure 2. We see a general trend where model performance improves as we increase the dimensionality of the fused representation. On the smaller Amazon Digital Music dataset, the performance increases up to dimension 256 and then starts to drop. This can be potentially attributed to over-fitting as there isn't enough data to properly utilize the increase in model capacity. On the other hand, on the larger Amazon CDs and Vinul dataset the performance continues to improve even at 512 dimensions although at a smaller rate.

4.6 Ablation Study

To quantify the contribution of each module in our architecture we conduct an extensive ablation study. We start with the BASE model which is a preference encoder with a single reconstruction decoder that optimises the MSE objective. This model is similar to AutoRec [20] with the addition of dropout on inputs to improve regularization. Next we consider BASE + NCE, where second decoder head is added to the model with the NCE objective. In parallel, we also consider BASE + REVIEW, where review encoder is added to the model. Note that in BASE + REVIEW we don't apply fusion and instead concatenate the outputs of the two encoders. Finally, in BASE + NCE + REVIEW we add both review encoder and NCE decoder head but still with concatenation instead of fusion. The difference between the full TAFA model and BASE + NCE + REVIEW is thus the addition of early and late stage fusion modules that combine the two modalities.

NDCG results for each model on four datasets are shown in Figure 3. We see that the addition of the NCE head in the BASE + NCE model significantly improves NDCG across all datasets, increasing performance by 2 to 4 points. Adding review encoder instead of

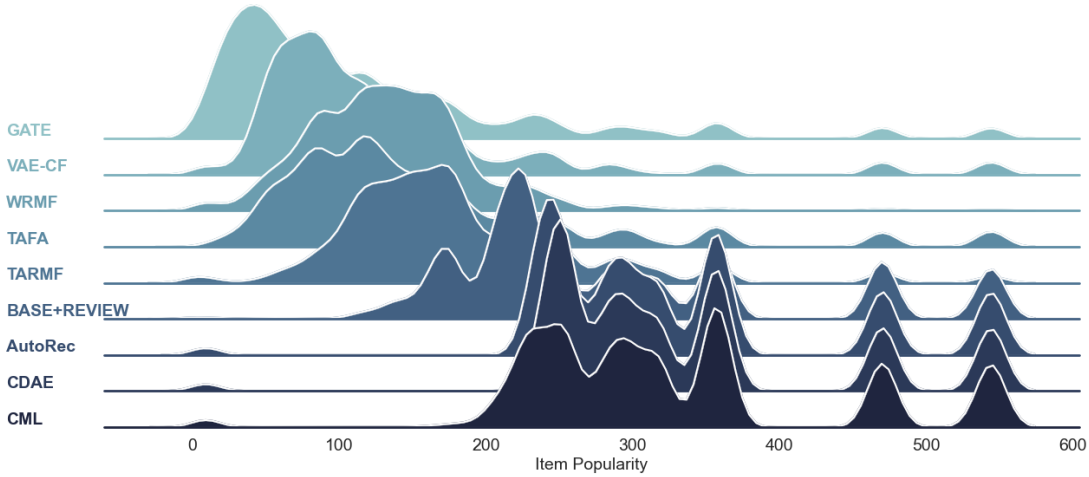
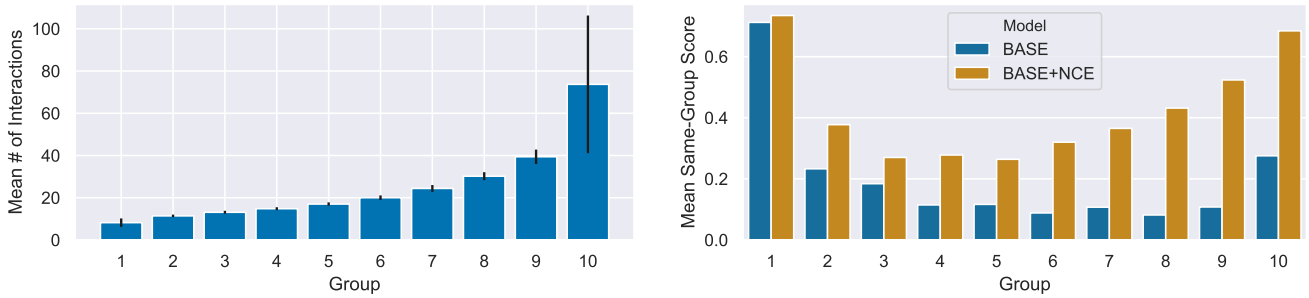


Figure 4: Amazon Video Games popularity distribution of the top-10 items recommended by each model across all users. X-axis represents item popularity (number of user interactions) and larger values indicate more popular items. Models are sorted according to the median of their popularity distribution. For our approach we show both the full TAFA model and ablated BASE + REVIEW model (see Section 4.6) which excludes the NCE decoder head.



(a) **(b)**

Figure 5: Yelp 2013 dataset, users are partitioned evenly into ten groups by number of interactions. Figure 5a shows the average number of interactions for each group. Figure 5b shows the average same-group retrieval score for each group. The score is computed by querying top-10 closest neighbors for each user using the Euclidean distance in the preference encoder embedding space e_u. We then compute the percentage of retrieved neighbors that are in the same group as the query user, and average these percentages across all users in each group.

NCE head in the BASE + REVIEW model gives an even larger gain particularly on the much sparser Amazon datasets. This further validates our hypothesis that the additional preference information from reviews can be highly beneficial in the sparse setting. Combining NCE head and review encoder in the BASE + NCE + REVIEW model provides further gain in performance which is amplified by the early and late stage fusion in the full TAFA model. The gain from fusion indicates that concatenation is not optimal for combining representations from multiple modalities, and properly fusing them is important.

4.7 Personalization

A good indicator of the level of personalization that a model provides is the popularity of items that it recommends. Figure 4 shows the popularity distribution of the top-10 recommended items by every model across all users on the Amazon Video Games dataset. Here, X-axis represents number of user interactions for each item, and larger values indicate more popular items. We see that AutoRec, CDAE and CML exhibit strong popularity bias and achieve relatively poor performance. In contrast, other baselines such as GATE and VAE-CF, mainly recommend less popular items with sharp peaks around very unpopular items. TAFA on the other hand, instead of being extreme on either end of the popularity spectrum, diversifies its recommendations in a more even way and is able to capture user preference for both popular and unpopular items. The balance between recommending popular and unpopular items can thus be one of the key reasons for strong performance of TAFA.

In addition to TAFA, we also plot popularity distribution for the ablated BASE + REVIEW model (see Section 4.6) that excludes the NCE decoder head. We see that, compared to the preference only

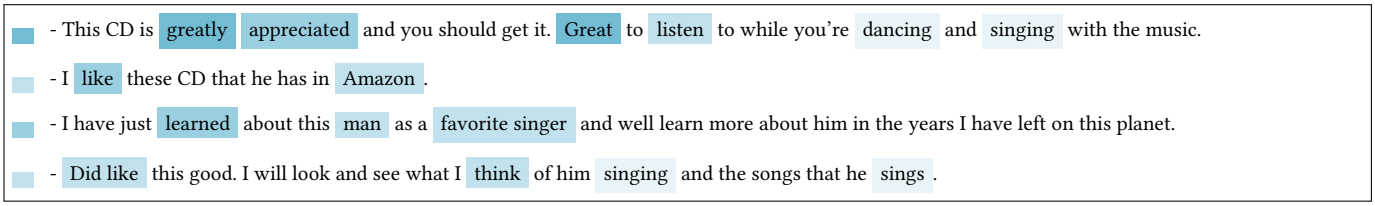


Figure 6: Amazon Digital Music short reviews from the same user. We show both word and review level (colored box next to each review) attention weights for each review. Darker color represents larger attention weights.

autoencoder model AutoRec, BASE + REVIEW shifts its recommendations towards less popular items. This indicates that incorporating review information makes the model more personalised. At the same time, we also observe that by adding the NCE decoder head, the full TAFA model shifts even more towards less popular items as compared to BASE + REVIEW and significantly improves performance as shown in Figure 3. This indicates that the NCE loss can effectively de-popularise the model further and improve the quality of recommendations.

4.8 Embedding Visualization

To provide further insight into the NCE objective and why it improves recommendations, we analyse the embedding space v_{fused} learned with and without the NCE decoder. As mentioned earlier, we hypothesize that the NCE objective can better regularize the embeddings by de-popularizing them. To verify this hypothesis we cluster users in the learned embedding space and then analyze the clusters. Recent work has found that the number of interactions in user profile correlates directly with preference for popular items [11]. Consequently, we first evenly partition users into ten groups based on the number of items they interacted with. We use the Yelp 2013 dataset and the average number of interactions in each user group is shown in Figure 5a. There is a significant variation between the groups with fewer than 10 interactions on average in the first group and more than 70 in the last one.

After grouping, we query the top-10 closest neighbors for each user using the Euclidean distance in the preference encoder embedding space e_u. We then compute the percentage of retrieved neighbors that are in the same group as the query user, and average these percentages across all users in each group. This is equivalent to computing user-based retrieval Precision@10 for each group. To remove the effects of other components we use the ablated models BASE and BASE + NCE (see Section 4.6) that only have the preference encoder. Note that for both models the fused embedding v_{fused} reduces to the preference encoder user embedding e_u and no review information is used. The results for this experiment are shown in Figure 5b. We see that embeddings from BASE + NCE have significantly higher same-group retrieval rates for all groups except the first one. First group contains near cold start users and both methods are able to successfully cluster them together. The middle groups have relatively low scores since the difference in the number of interactions is not significant between them as seen from Figure 5a. However, BASE + NCE scores are still considerably higher than BASE alone with more than 3x increase for some groups. Finally, similar pattern is observed for the upper groups where BASE + NCE again improves in every group, and in particular is able to accurately cluster most active users in the last group. These results indicate that adding the NCE decoder head enables the model to learn embeddings that better cluster users by the level of activity which improves personalisation.

4.9 Qualitative Analysis

By tracing the attention weights back to review text we can visualise parts that the model is focusing on. This can be used to explain generated recommendations and gain further insight into user preferences. Our model has both review and word level attention. Combined, the two attention modules provide both global view into reviews that the model finds useful and local view on specific information that is used from each review. We show both types of attention in Figure 6. The reviews are taken from the Amazon Digital Music dataset and belong to the same user. The colors represent attention weights with darker color indicating larger attention weight.

We see that for review level attention the model is focusing more on reviews 1 and 3 than 2 and 4. Both reviews 1 and 3 are longer and have specific preference reasons whereas, reviews 2 and 4 are relatively vague and ambiguous. In particular, in review 4 the reviewer expresses an undecided opinion about an artist. We also see that for word level attention the model focuses on words such as "great(ly)", "favorite" and "like" that express direct preference. In review 1 some attention is also put on the reasons why the reviewer likes a particular CD such as "singing" and "dancing". This example demonstrates that by combining review and word level attention the model can effectively extract relevant information from multiple user reviews that can be ambiguous and noisy.

While short reviews are relatively easy to model, longer review pose a bigger challenge as they often contain multiple opinions that need to be distilled. Figure 7 shows word level attention on three longer reviews from different users on the Amazon Digital Music dataset. We still see that the model is able to pick up informative preference words such as "wonderful", "pleasing", "nice" and "fresh" throughout the reviews. In addition, some of the attention is focused on the target artist in each review such as "Mr Neil Diamond" and "Teedra". Combining this information the model can infer preference for these artists and incorporate it into user embedding to improve recommendations.

5 CONCLUSION

In this work, we present a novel two-headed attention fusion autoencoder model that leverages both user-generated reviews and implicit feedback to make recommendations. We introduce novel

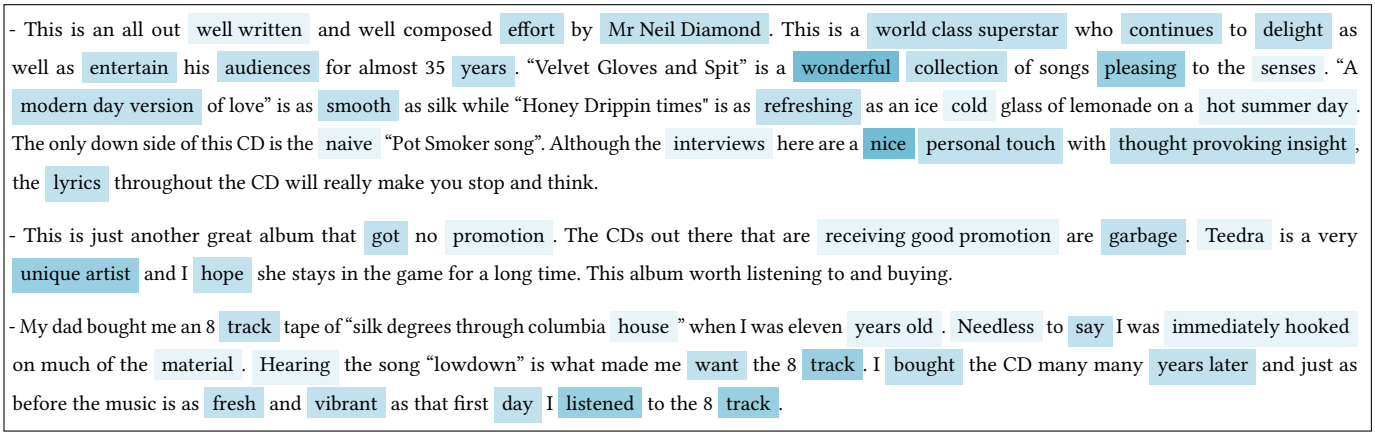


Figure 7: Amazon Digital Music examples of long reviews from different users with word level attention. Darker color represents larger attention weights.

multi-stage fusion module to combine the information from the two modalities, and utilize two-headed decoder with MSE and NCE losses to decode the fused representation. Extensive ablation study demonstrates that each added component improves performance, and the NCE decoder head effectively de-popularises the model making it effective for both popular and unpopular items. Extensive empirical evaluation on six real-world datasets from Yelp and Amazon show that our approach consistently outperforms leading baselines. Future works involves additional investigation into the NCE objective and other data sources such as item images, video and audio.

REFERENCES

[1] Alejandro Bellogín, Pablo Castells, and Iván Cantador. 2017. Statistical Biases in Information Retrieval Metrics for Recommender Systems. *Information Retrieval Journal* 20, 6 (2017), 606–634.

[2] Rocío Cañamares and Pablo Castells. 2018. Should I Follow the Crowd? A Probabilistic Analysis of the Effectiveness of Popularity in Recommender Systems. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*.

[3] Rose Catherine and William Cohen. 2017. Transnets: Learning to transform for recommendation. In *Proceedings of the 11th ACM conference on Recommender Systems*.

[4] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural attentional rating regression with review-level explanations. In *Proceedings of the 27th International Conference on World Wide Web*.

[5] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems*.

[6] Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*.

[7] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. (2017). arXiv: 1708.05031.

[8] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. 2017. Collaborative Metric Learning. In *Proceedings of the 26th International Conference on World Wide Web*.

[9] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. 2016. Convolutional matrix factorization for document context-aware recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*.

[10] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* (2009), 30–37.

[11] Dominik Kowald, Markus Schedl, and Elisabeth Lex. 2020. The Unfairness of Popularity Bias in Music Recommendation: A Reproducibility Study. In *European Conference on Information Retrieval*. Springer, 35–42.

[12] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *Proceedings of the 27th International Conference on World Wide Web*.

[13] Yichao Lu, Ruihai Dong, and Barry Smyth. 2018. Coevolutionary Recommendation Model: Mutual Learning between Ratings and Reviews. In *Proceedings of the 27th International Conference on World Wide Web*.

[14] Chen Ma, Peng Kang, Bin Wu, Qinglong Wang, and Xue Liu. 2019. Gated attentive-autoencoder for content-aware recommendation. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*.

[15] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *Proceedings of the 8th IEEE International Conference on Data Mining*.

[16] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*.

[17] Alan Said and Alejandro Bellogín. 2014. Comparative recommender system evaluation: benchmarking recommendation frameworks. In *Proceedings of the 8th ACM Conference on Recommender Systems*.

[18] Badrul M Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2002. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proceedings of the 5th International Conference on Computer and Information Technology*.

[19] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Darius Braziunas. 2016. On the effectiveness of linear models for one-class collaborative filtering. In *30th AAAI Conference on Artificial Intelligence*.

[20] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. AutoRec: Autoencoders Meet Collaborative Filtering. In *Proceedings of the 24th International Conference on World Wide Web*.

[21] Florian Strub and Jeremie Mary. 2015. Collaborative filtering with stacked denoising autoencoders and sparse inputs.

[22] Yunzhi Tan, Min Zhang, Yiqun Liu, and Shaoping Ma. 2016. Rating-Boosted Latent Topics: Understanding Users and Items with Ratings and Reviews. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*.

[23] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and Composing Robust Features with Denoising Autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*.

[24] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*.

[25] Ga Wu, Maksims Volkovs, Chee Loong Soon, Scott Sanner, and Himanshu Rai. 2019. Noise Contrastive Estimation for One-Class Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[26] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the 9th ACM International Conference on Web Search and Data Mining*.

[27] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)* 52, 1 (2019), 1–38.

[28] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining*.

[29] Jin Peng Zhou, Ga Wu, Zheda Mai, and Scott Sanner. 2020. Noise Contrastive Estimation for Autoencoding-based One-Class Collaborative Filtering. arXiv:2008.01246 [cs.IR]