MARec: Metadata Alignment for cold-start Recommendation

Julien Monteil
Amazon Machine Learning
Brisbane, Australia

Volodymyr Vaskovych
Amazon Machine Learning
Melbourne, Australia

Wentao Lu
Amazon Machine Learning
Sydney, Australia

Anirban Majumder
Amazon Machine Learning
Bangalore, India

Anton van den Hengel
University of Adelaide
Adelaide, Australia

ABSTRACT

For many recommender systems, the primary data source is a historical record of user clicks. The associated click matrix is often very sparse, as the number of users × products can be far larger than the number of clicks. Such sparsity is accentuated in cold-start settings, which makes the efficient use of metadata information of paramount importance. In this work, we propose a simple approach to address cold-start recommendations by leveraging content metadata, Metadata Alignment for cold-start Recommendation (MARec). We show that this approach can readily augment existing matrix factorization and autoencoder approaches, enabling a smooth transition to top performing algorithms in warmer set-ups. Our experimental results indicate three separate contributions: first, we show that our proposed framework largely beats SOTA results on 4 cold-start datasets with different sparsity and scale characteristics, with gains ranging from +8.4% to +53.8% on reported ranking metrics; second, we provide an ablation study on the utility of semantic features, and proves the additional gain obtained by leveraging such features ranges between +46.8% and +105.5%; and third, our approach is by construction highly competitive in warm set-ups, and we propose a closed-form solution outperformed by SOTA results by only 0.8% on average.

ACM Reference Format:
Julien Monteil, Volodymyr Vaskovych, Wentao Lu, Anirban Majumder, and Anton van den Hengel. 2024. MARec: Metadata Alignment for cold-start Recommendation. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Recommender systems have been a hot topic of interest in the research community, even more so since we interact with large-scale recommender systems in our daily lives. Significant effort has been dedicated to surpass the performance of state-of-the-art methods on public datasets, focusing on learning the best representation of the sparse user-item interaction matrix. Less interest has been vested on solving the different yet practical problem of cold-start recommendation which consists of providing relevant recommendations for new users and items as soon as they appear on the platform.

Cold-start is a significant challenge for recommender systems as traditional collaborative filtering techniques rely heavily on past user interactions which are not available for new users or items. As a result, existing systems that are optimized for warm (previously seen) users may generalize poorly, if at all, to the cold-start scenario. As recommender systems constantly experience flurry of new items and users, high quality cold-start recommendations are essential for a great user experience and long-term engagement. It is important to highlight that the cold-start scenario extends beyond just looking at the ranking performance on new items (or users); relevant recommendations are required for the mixed scenarios as well i.e. with a blend of warm and cold items (users) with varying amount of historical data.

Amongst early works on cold-start recommendation modeling figure the works of [15, 29] which respectively map metadata features to Matrix factorization latent factors, and leverage item metadata reconstruction for regularization. More recent techniques for cold-start modeling include meta learning to combine user history with item representation [42], Variational AutoEncoders [47], and the novel idea of content-aware hashing [48]. However, while current approaches for cold-start modeling demonstrate good performance in cold-start scenarios, their effectiveness tends to decline significantly when applied to warm-start settings. In an industrial setup, managing separate models for cold-start and warm-start proves cumbersome. This not only adds to the overhead of maintaining multiple models in production but also poses a challenge to ensure a consistent and high-quality pool of recommendations for users when they transition between the cold and warm regime. In this paper, we present MARec (**M**etadata **A**lignment for cold-start **Rec**ommendation), a new approach that excels beyond current methods in cold-start recommendation all the while being competitive with state-of-the-art (SOTA) techniques for warm-start recommendation.

Our approach builds upon the existing body of literature in particular the hybrid approaches of merging content-based and collaborative filtering techniques [21, 23] as well as the recent works on retrieval-augmented systems [7]. Our core contribution is a novel yet simple idea of aligning item-item similarities estimated from user clicks and from the item metadata features via a regularization term that pushes the cold-start items onto the same similarity space. In this way, we leverage SOTA models for warm-start recommendation to learn an alignment of the feature-based

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

similarities to the user-item interaction similarities, which provides a high-performing cold-start recommender.

1.1 Contributions

To the best of our knowledge, the alignment framework we propose differs from the existing body of literature, as it can augment any top performing Matrix Factorization and (variational) autoencoders approaches, it makes the most of any additional metadata features we can leverage across modalities (image, categorical, textual), and yet it achieves impressive results despite its simplicity. We present the following contributions:

(1) We present a novel algorithm (MARec) for cold-start recommendation that achieves state-of-the-art performance on several benchmarking datasets. Our algorithm combines embeddings learned from item and customer metadata with the user-item click matrix and gives closed-form update equations.
(2) We show that MARec beats SOTA techniques on four cold-start benchmarking datasets with different sparsity and scale characteristics, with gain ranging from +8.4% to +53.8% on standard ranking metrics. MARec achieves this performance while being orders of magnitude faster, in terms of training time, compared to the best performing baseline.
(3) We provide a study on the utility of Large Language Models (LLMs) embeddings and demonstrate that the additional gain obtained by leveraging semantic features ranges between +46.8% and +105.5%.
(4) Finally, MARec enables a transition to near-SOTA performance in warm set-ups, and we introduce a closed-form solution outperformed by SOTA results on warm datasets by only 0.8% on an average.

2 RELATED WORKS

The historical way to perform collaborative filtering with click datasets is via neighbourhood methods which consists of computing similarities across items or users, where the similarity is evaluated as a dot product based on click behaviour, and possibly item and user metadata [18]. The Netflix competition and subsequent research works have shown that linear and matrix factorization techniques tend to outperform neighbourhood methods [22, 28, 40]. A push for neural approximations of matrix factorization took place [11, 17], with the conclusion that the proposed architectures fail to learn a better nonlinear variant of the dot product and are outperformed by careful implementations of matrix factorization [12, 35, 36]. More recently, the variational autoencoder approach [24] was proposed, but the least squares approximation of SLIM [28], which is a linear autoencoder with projections to ensure the zero diagonal constraint (EASE) [38, 40], still beat it on 2 out 3 public datasets. It was shown in [20] that augmenting the VAE approach with flexible priors and gating mechanisms led to SOTA performance. The VAE approach was also refined and ensembled with a neural EASE to achieve SOTA [41] on MovieLens20M and Netflix. The success of Deep Cross Networks on learning to rank tasks [32] also showed the importance of explicitly encoding the dot product in the network architecture.

Comparatively, less research has been done on the usefulness of the SOTA collaborative filtering algorithms for cold-start recommendations. An early effort to incorporate item metadata into the SLIM family of models was the one of [29], where it is shown that adding a regularization term pushing towards the reconstruction of the feature information matrix, could help achieve gain on some datasets. Early methods addressed this fusion problem by proposing hybrid models merging neighborhood and collaborative filtering methods [21, 23], where the core idea relies on weighting a content-based filtering term and a collaborative filtering term in the objective function, thereby training the model to leverage content representations when preferences are not available. In [31], the sparsity problem is mitigated for cold start by developing a model at the word level, not at the document level. More recently, it was shown in [9] that utilising different regularization weights to the latent factors associated to users and items shows consistent benefits for addressing cold-start in matrix factorization. In [43], a dropout technique is leveraged to set some click inputs to zero in order to force the model to reconstruct the relevance scores without seeing the warm inputs. A meta learning algorithm deployed in a production system was proposed in [42] to combine user history representation with item representation.

Another recent work [46] relies on maximizing the mutual information between feature representations and user-item collaborative embeddings, and it is shown that the presented technique beats a number of previous approaches [14, 16] building on the same intuition. This approach was extended in [45], which addresses the item feature shift generated by cold-start items via an equivariant learning framework. The variational framework of [24] is adapted in [47] for addressing cold-start in particular to obtain reasonable priors so that to IDs with the similar attributes can naturally cluster together in the latent embedding space. In [48], a content-aware hashing approach named Discrete Deep Learning (DDL), which first relies on building an user-item preference model based on hash codes, is proposed to alleviate data sparsity and cold-start item problem. A hybrid model is proposed in [6] where a neural model is trained to learn feature embeddings and item similarities from the similarities pre-learned from a collaborative filtering model, e.g. SLIM. In [49], a method is proposed to warm-up cold item embeddings by modelling the relationship between item id and side information in the latent space.

The ensembling of complementary models remains another possibility for boosting cold-start recommendations, e.g., the winning solution of the 2018 ACM challenge on a cold-start music recommendation dataset was the score ensembling of a content-aware deep cross network and a gradient boosted decision tree [5]. Finally, retrieval-augmented models present similar architectures to hybrid recommender systems, e.g. in [19] where the linear interpolation of a pre-trained LLM with a k-nearest neighbors model led to SOTA results on Wikitext-103, in [7] where they use a chunked cross-attention mechanism to condition the training of the language model to encoded neighbours, and in [25] where a retrieval module to return the top-k text descriptions associated to the image helps achieves SOTA results for visual long tail classification.

In the context of the above literature, our paper situates itself in the category of hybrid recommender systems, at the intersection of collaborative and content-based filtering. Differently to the existing literature on hybrid recommender systems, we put a specific focus on achieving scalability, and on leveraging the benefits of collaborative filtering techniques, as we transition to warmer situations. Typical hybrid approaches tend to complicate the learning by forcing the balancing of multiple objective terms, in addition to training content representations. In particular, the content part of the objective function is trained to explain the observed data, hence it may model content aspects that end up hurting cold-start recommendations. In our framework, we provide a flexible scalable approach that learns to only leverage the relevant features of content metadata, and aligns the notion of item similarities independently of whether it comes from click data or item metadata.

3 MODELLING APPROACH

Let $\mathcal{U}$ and $\mathcal{I}$ denote the set of users and items respectively. We assume that the data is available as a sparse matrix $\mathbf{X} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$ where X_{ij} denotes the interaction (click, rating) between user $i \in \mathcal{U}$ and item $j \in \mathcal{I}$. A value of 1 in X_{ij} indicates that the user interacted with the item while a value of 0 implies that no interaction has been observed. We use the shorthand notation $\mathbf{X_{i.}}$ and $\mathbf{X_{.j}}$ to denote the i^{th} row and j^{th} column of $\mathbf{X}$. Similarly, we use x_i to denote the i^{th} element of a vector x.

For each item, we have access to a list of N attributes representing metadata associated with each item such as item name, description, brand, etc. We assume that the k^{th} attribute across all items has a compact feature representation $\mathbf{F^{(k)}} \in \mathbb{R}^{|\mathcal{I}|\times n_k}$ where n_k is the dimension of the feature space. Finally, $\mathbf{F} \in \mathbb{R}^{|\mathcal{I}|\times K}$ represents the feature vector of items with $K = \sum_{k=1}^{N} n_k$.

3.1 MARec

MARec consists of three components: 1) a backbone model f^{B} which is a latent network that learns a low-dimensional representation of the interaction data $\mathbf{X}$ and its reconstruction, 2) an embedding model f^{E} that encodes a dense representation of the item metadata and 3) an alignment model f^{A} that aligns the metadata representation with the click history. This aligned representation can then be fused with the backbone model to reconstruct the click history. The backbone model $f^{\text{B}} : \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|} \to \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$ can be chosen among the well-performing collaborative filtering algorithms of matrix factorization and autoencoder families, e.g., [20, 38]. Below we describe each component in detail.

The embedding model $f^{\text{E}} : \mathbb{R}^{|F|\times|\mathcal{I}|} \to \mathbb{R}^{|E|\times|\mathcal{I}|}$ learns a dense representation of item metadata so that items with similar interaction pattern are mapped closer in the embedding space. The alignment model $f^{\text{A}} : \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|} \times \mathbb{R}^{|E|\times|\mathcal{I}|} \to \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$ is responsible for aligning interaction data with the item-item similarity estimated via the embedding model f^{E}. Figure 1 presents the high-level overview of MARec.

The model is learned by minimizing a weighted combination of the backbone model loss and the alignment loss:

$$\min_{\theta,\phi,\zeta} L_{\text{A}}\left(\mathbf{X}, f^{\text{B}}_{\theta}(\mathbf{X})\right) + L_{\text{B}}\left(f^{\text{B}}_{\theta}(\mathbf{X}), f^{\text{A}}_{\zeta}\left(\mathbf{X}, f^{\text{E}}_{\phi}(\mathbf{F})\right)\right) \tag{1}$$

where θ, ϕ, ζ are the model parameters of individual components of MARec. Note that L_{A}, L_{B}, can be any loss function, e.g., quadratic,

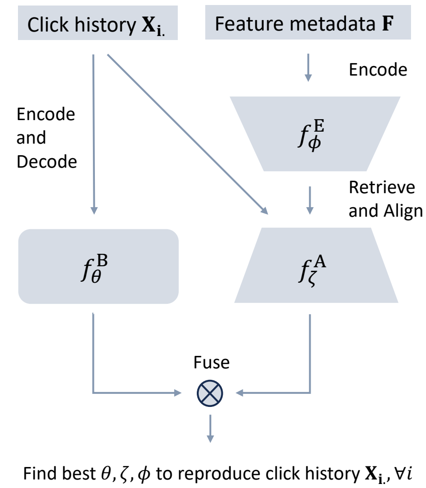

Figure 1: Simplified architecture of MARec. The embedding model $f^{\mathbf{E}}_{\phi}$ embeds each item metadata in a joint embedding space, and the alignment function $f^{\mathbf{A}}_{\zeta}$ retrieves and align the similar items to the similarity space provided by the backbone model $f^{\mathbf{B}}_{\theta}$. The architecture is trained with a reconstruction loss on the click history.

cross entropy, cosine, multinomial likelihood etc., possibly augmented with regularization terms and constraints, as in the case of autoencoders to prevent overfitting [39]. While the first term of the loss function is the familiar one leveraged in matrix factorization and autoencoder techniques, the second term can be understood as a regularization to push the collaborative filtering similarities towards the learned metadata embeddings. This is particularly beneficial for cold-start items as they are under-represented in the interaction data. It is to be noted that MARec modifies only the training objective by adding a regularization term. Once trained, the backbone model can be used for inference as usual. MARec can be seen as a generalization of the merging of collaborative and content-based filtering techniques and admits a closed-form solution with appropriate choice of the components.

3.2 Embedding Model

The embedding model can be either the identity function $f^E = \mathbf{I}$, keeping the encoded metadata features $\mathbf{F}$ as they are, with $|E| = |F|$, or a Siamese network [8] trained on the processed click data $\mathbf{X}$ which aims at learning joint metadata embeddings. We follow a similar process to [27] to train the Siamese network. For each item $I_j \in \mathcal{I}$, we select k items with the highest cosine similarity $\mathbf{X_{.j}}^T\mathbf{X}$ and assign them a label of one. Similarly, we sample k items with the lowest cosine similarities and label them as zero. This gives us a dataset of similar and dissimilar pairs of the form $\left(\mathbf{F_{.j}}, S_j^+, S_j^-\right)$ where S_j^+ is the set of positive pairs and S_j^- is the set of negative pairs. We then train the network by minimizing the contrastive loss [44]:

$$\sum_j \sum_{k^+ \in S_j^+} \left[-\log \frac{\exp\left(z_{k^+}^T \cdot z_j / \tau\right)}{\exp\left(z_{k^+}^T \cdot z_j^T / \tau\right) + \sum_{k^- \in S_j^-} \exp\left(z_{k^-}^T \cdot z_j^T / \tau\right)} \right] \quad (2)$$

where $z_j = f^{\mathrm{E}}(\mathbf{F}_{.\mathbf{j}}), z_{j^+} = f^{\mathrm{E}}(\mathbf{F}_{.\mathbf{j}^+}), z_{j^-} = f^{\mathrm{E}}(\mathbf{F}_{.\mathbf{j}^-})$ are the latent representations learned by the embedding model and $\tau > 0$ is a temperature hyper-parameter. The embedding layer presents the benefit of directly optimizing for the (dis)similarities of items which in turn is leveraged by the alignment model.

3.3 Alignment Model

The alignment layer is a critical component of MARec with the objective of aligning the observed click matrix $\mathbf{X}$ with an item-item similarity measure defined over the metadata embeddings encoded by the embedding layer f^{E}. The item-item similarity is typically captured via a smoothed cosine transform [12] $g : \mathbb{R}^{|\mathcal{I}| \times |E|} \to \mathbb{R}^{|\mathcal{I}| \times |\mathcal{I}|}$,

$$g\left(f^{\mathrm{E}}(\mathbf{F})\right) = \mathbf{G}, \quad G_{ij} = \frac{z_i^T \cdot z_j}{\|z_i\| \cdot \|z_j\| + \delta}, \quad i, j \in \mathcal{I} \quad (3)$$

where $\delta > 0$ is a smoothing term used to lower the similarity between items with very sparse feature vectors [12]. When the embedding model is the identity function, the smoothed cosine transform simply computes similarities by taking the complete feature vector $\mathbf{F}$ as input, and this is known as itemKNN [13]. However, it is expected that some attributes in the metadata are more predictive of item similarity than others. Hence an alternative is to compute embedding of each attribute $\mathbf{F}^{(\mathrm{k})}$ separately, and then weigh attribute representations at a later stage. More specifically, starting from the encoded representation of the k$^{\text{th}}$ attributes, $z_j^{(k)} = f^{\mathrm{E}}(F_{.j}^{(k)})$, we define item-item similarity based on the k$^{\text{th}}$ attribute as

$$G_{ij}^{(k)} = \frac{z_i^{(k)} \cdot z_j^{(k)}}{\|z_i^{(k)}\| \cdot \|z_j^{(k)}\| + \delta}, \quad i, j \in \mathcal{I}, \quad (4)$$

which leads to N similarity matrices mixed via a quadratic form:

$$g\left(f^{\mathrm{E}}(\mathbf{F})\right) = \sum_{k<l} \left(\mu_k \cdot \mathbf{G}^{(\mathbf{k})} + \mu_{kl} \cdot \mathbf{G}^{(\mathbf{k})} \cdot \mathbf{G}^{(\mathbf{l})}\right), \quad (5)$$

where the scalar coefficients $\mu_k \geq 0$ and $\mu_{kl} \geq 0$ are model parameters that learn importance of first and second order similarities across the feature sub-spaces. This quadratic form explicitly model non-linear similarities across feature vectors, e.g. between actors, directors, genres, in the case of movie recommendation. Finally, we define the alignment function as,

$$f^{\mathrm{A}}\left(\mathbf{X}, f^{\mathrm{E}}(\mathbf{F})\right) = \alpha \cdot \mathbf{X} \cdot g\left(f^{\mathrm{E}}(\mathbf{F})\right) \cdot \mathbf{D}^{\mathbf{R}} \quad (6)$$

where $\alpha \in \mathbb{R}_+^*$ is a scaling coefficient, $\mathbf{D}^{\mathbf{R}} = [d_j]_{j \in [|\mathcal{I}|]} \in \mathbb{R}_+^{|\mathcal{I}| \times |\mathcal{I}|}$ is a regularizing diagonal matrix that weights the importance of the metadata embeddings as a function of the number of clicks per item. The entries in $\mathbf{D}^{\mathbf{R}}$ are set as $d_j = h\left(\sum_j \mathbf{X}_{.\mathbf{j}}\right)$, with $h : \mathbb{N} \to \mathbb{R}^+$, a non-negative monotone decreasing linear or exponential function. Note that a monotone decreasing function is used to mitigate the popularity bias problem [50], thereby giving more importance to cold-start items during model training. The significance of popular items is nevertheless captured in the backbone model that we describe next.

3.4 Backbone Model

Our framework supports any model of the autoencoder and matrix factorization families as we aim to align the item embeddings with the learned similarities from the click matrix itself. We consider 3 competitive baselines in this work, the multi-VAE model of [24], a collective version of the EASE model [29, 38], and a modified SLIM model that we introduce.

Variational autoencoders. For each user $i \in \mathcal{U}$, the model samples a K-dimensional latent representation ξ_i from a standard Gaussian prior. The latent representation ξ_i is then passed to a multilayer perceptron $f_\theta(\xi_i)$ to produce a probability distribution over the set of items $\pi(\xi_i)$ from which the click history $\mathbf{X}_{\mathbf{i}.}$ is assumed to have been drawn. For each user i, we then adapt the multinomial log-likelihood to be maximized and the global objective of equation (1) can be modified to:

$$\begin{aligned} \min_\theta -\log p_\theta(x_i|\xi_i) &= \sum_{j \in \mathcal{I}} -X_{ij} \log(\pi_j(\xi_i)) - b_{ij} \log(\pi_j(\xi_i)), \\ \mathbf{b_i} &= f^{\mathrm{A}}\left(\mathbf{X}_{\mathbf{i}.}, f^{\mathrm{E}}(\mathbf{F})\right). \end{aligned} \quad (7)$$

To learn the parameters θ of the generative model, we need to approximate the intractable posterior distribution $p(\xi_i|X_{i.})$ and may use variational inference as in [20, 24, 47]. We note that the second term added to the objective takes care of both the alignment and fusion by adapting the learned probability distribution over items to the metadata-based estimates. Here, the multi-VAE model of [24] was considered but we can readily consider its recent enhancements, H+Vamp [20] and VASP [41], which now occupy the first 2 slots of the MovieLens20M and Netflix leaderboards.

EASE. We adapt the EASE model [38] to our global objective of equation (1) which becomes:

$$\begin{aligned} &\min_\Theta \|\mathbf{X} - \mathbf{X\Theta}\|_F^2 + \lambda_1 \|\mathbf{\Theta}\|_F^2 + \|\mathbf{X\Theta D^R} - f^{\mathrm{A}}\left(\mathbf{X}, f^{\mathrm{E}}(\mathbf{F})\right)\|_F^2 \\ &\text{s.t.} \quad \mathrm{diag}(\mathbf{\Theta}) = 0, \end{aligned} \quad (8)$$

with $\mathbf{\Theta} \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{I}|}$ and $\beta \in \mathbb{R}$ a scalar coefficient. Note that we may add the collective term $\lambda_0 \|\mathbf{F} - \mathbf{F\Theta}\|_F^2$ to the objective function as in [29]. This optimization policy admits a closed formed solution via the use of a Lagrangian multiplier heuristics that sets diag($\mathbf{\Theta}$) to zero [38] and the solution is written:

$$\begin{aligned} \mathbf{P} &= \left(\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda_0 \mathbf{F}^{\mathrm{T}}\mathbf{F} + \lambda_1 \mathbf{I} + \mathbf{X}^{\mathrm{T}} f^{\mathrm{A}}\left(\mathbf{X}, f^{\mathrm{E}}(\mathbf{F})\right)\right)^{-1}, \\ \tilde{\mathbf{\Theta}} &= \mathbf{P}\left(\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda_0 \mathbf{F}^{\mathrm{T}}\mathbf{F} + \mathbf{X}^{\mathrm{T}} f^{\mathrm{A}}\left(\mathbf{X}, f^{\mathrm{E}}(\mathbf{F})\right)\right), \\ \mathbf{\Theta} &= \tilde{\mathbf{\Theta}} - \mathbf{P}\frac{\mathrm{diag}(\tilde{\mathbf{\Theta}})}{\mathrm{diag}(\mathbf{P})}. \end{aligned} \quad (9)$$

Modified SLIM. We do observe that the least squares framework of [38] could be further improved for two reasons: first, by giving less weight to negative samples, as we observe that matrix factorization results in [35] were boosted by a simple negative subsampling strategy; second, by relaxing the projection heuristics used to enforce the $\mathrm{diag}(\mathbf{\Theta}) = 0$ constraint. We can then write a constraint-free optimisation policy and the backbone model loss of equation (1) becomes:

$$\min_{\mathbf{\Theta}} \sum_i \|\mathbf{W_i}^{\frac{1}{2}} (\mathbf{X_{.i}} - \mathbf{X\Theta_{.i}})\|_F^2 + \lambda_1 \|\mathbf{\Theta_{.i}}\|_F^2 + \gamma_1 |\Theta_{ii}|^2, \quad (10)$$

where $\mathbf{W_i} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{U}|}$, $i \in \mathcal{I}$, are the diagonal matrices of the weights w_0 and w_1 for positive and negative samples, and $\gamma_1 \sum_i |\Theta_{ii}|^2$ is the new regularization term that relaxes the constraint $\mathrm{diag}(\mathbf{\Theta}) = 0$ but serves the same goal of penalising the trivial $\mathbf{\Theta} = \mathbf{I}$ solution. The closed-form solution to the global objective of equation (1) is now written, $\forall i \in \mathcal{I}$:

$$\begin{aligned} \mathbf{\Theta_{.i}} &= \left(\mathbf{X}^{\mathrm{T}}\mathbf{W_i}\mathbf{X} + \mathbf{X}^{\mathrm{T}}\mathbf{W_i}\mathbf{B} + \lambda_1 \mathbf{I} + \mathbf{\Gamma^{(i)}}\right)^{-1} \left(\mathbf{X}^{\mathrm{T}}\mathbf{W_i}\mathbf{X_{.i}} + \mathbf{X}^{\mathrm{T}}\mathbf{W_i}\mathbf{B_{.i}}\right), \\ \mathbf{B} &= f^{\mathrm{A}}\left(\mathbf{X}, f^{\mathrm{E}}(\mathbf{F})\right), \end{aligned} \quad (11)$$

with $\mathbf{\Gamma^{(i)}}$ the diagonal matrix made of zeros except at position i where $\Gamma_{ii}^{(i)} = \gamma_1$, and $\mathbf{B_{.i}}$ the i^{th} column of the alignment matrix.

4 EXPERIMENTS

We leverage cold and warm dataset splits used in previous works where SOTA results were reported, for reproducibility reasons.

4.1 Datasets

Cold splits. Experimental evaluation of MARec requires datasets with varying sparsity level, and where item metadata can be leveraged for cold-start recommendations. We select popular dataset splits explored in the cold-start recommendation literature [6, 46], as shown in Table 1.

	# users	# items	# item features	Interactions (%)
Amazon Video Games [6]	33858	12463	5	0.06
Netflix [3] [6]	469986	9503	6	1.19
MovieLens10M [2]	55485	5986	5	1.34
MovieLens Hetrec [6]	2107	6234	6	3.10

Table 1: Characteristics of cold-start data splits used in our experiments ordered by sparsity level.

The dataset splits were generated from binarized click interaction data to reproduce a cold-start item scenario as per the procedure described in the initial work of [37]. For the Amazon Video Games and MovieLens Hetrec datasets, we leverage the publicly available splits[1] of [6], for which 60% of items are kept for training, 20% for validation and 20% for test. The obtained metrics are averaged over 10 different random splits of train, validation and test sets. For the MovieLens10M and the Netflix datasets, 20% of cold-start items are sampled and they are split into the cold validation and test sets. Then we proceed with a random 80%, 10%, 10% split on the rest of the click matrix to get the train, warm validation and test sets. For the MovieLens10 dataset, the hybrid validation and test sets are obtained after combining the cold and warm validation and test sets. All models are trained on the training set with hyper-parameter optimization performed on the validation set.

Regarding item metadata, we take the same set of attributes used in [6, 46]. For Amazon Video Games, we use *title name, description, feature description, brand* and *categories.* For MovieLens Hetrec, we use *movie years, genres, actors, directors, countries,* and *locations.* For MovieLens10M, we use *title, description, genres, actors,* and *directors.* For Netflix, we use *title, genres, actors, directors, producers,* and *composers.* For the Netflix dataset, we join the titles of the Netflix prize dataset with the titles of the IMDb non-commercial dataset [1].

	# users	# items	Interactions (%)
Pinterest [17]	55187	9916	0.27
MovieLens1M [17]	6040	3706	4.47

Table 2: Characteristics of the warm data splits used in experiments ordered by sparsity level.

Warm splits. We also present our analysis on publicly available data splits[2] for warm-start recommendation that received a lot of attention in the literature [12, 17, 35]. The dataset characteristics are summarized in Table 2. The splits were generated by selecting users with at least 20 ratings and using the leave-one-out evaluation, ranking the last click together with 100 randomly drawn negative samples. The validation set is generated from the training set using the same protocol as in [35].

4.2 Metrics and statistical significance

We leverage standard metrics reported in recommendation literature [24] namely, hit rate (hr@k) and normalized discounted cumulative gain (ndcg@k). Given a validation or test set containing a set of users $\mathcal{U}' \subseteq \mathcal{U}$, let $\mathcal{I}_u \subseteq \mathcal{I}$ be the set of items clicked by user u. The metric hr@k is defined as:

$$\mathrm{hr@k} = \frac{1}{|\mathcal{U}'|} \sum_{u\in\mathcal{U}'} \frac{\sum_{i\in\mathcal{I}_u} \mathbb{I}(r_{ui} \le k)}{\min(k, |\mathcal{I}_u|)}, \quad (12)$$

where $\mathbb{I}$ is the indicator function and r_{ui} the predicted rank of item i for user u. The ndcg@k is defined as:

$$\mathrm{ndcg@k} = \frac{1}{|\mathcal{U}'|} \sum_{u\in\mathcal{U}'} \left(\sum_{i\in\mathcal{I}_u} \frac{\mathbb{I}(r_{ui} \le k)}{\log_2(1+r_{ui})} \Big/ \sum_{j\in K'} \frac{1}{\log_2(1+j)} \right), \quad (13)$$

where the set K' is defined as $K' = \{1, \dots, \min(k, |\mathcal{I}_u|)\}$.

In the rest of the paper, we report average gain observed across all users in the test datasets. We also performed bootstrapping analysis to assess the significance of our results by sampling 20% of the users in each test dataset and repeating the process 500 times, in order to compute 95% confidence intervals, and we indicate when our results achieve statistical significance.

[1] https://github.com/cesarebernardis/NeuralFeatureCombiner

[2] https://github.com/hexiangnan/neural_collaborative_filtering

4.3 Results for Cold Splits

We now report ndcg@k and hr@k metrics on the cold data splits. We first focus on incorporating item features with the same featurization policy as in [6, 46]. A study into more sophisticated ways to encode the item metadata features is also conducted. We discuss embeddings and alignment strategies to achieve the best results with item metadata features only. Finally, we explore the incorporation of more item metadata features and discuss the lifts obtained. Regarding the baselines considered, we largely make use of the code repository[3] of [6] for hyper-parameter tuning, training and scoring. Hyper-parameter optimization was done via grid search on the validation sets, following the exact same process across all models. We train the baselines and our proposed approach on an ml.p3.2xlarge EC2 instance.

Baselines. We report the results for several competitive baselines: NFC [6], ItemKNNCF [26], Wide and Deep [10], and FM [34]. NFC is the hybrid model of [6], which was reported to largely beat SOTA for cold item recommendations. ItemKNNCF is an item content-based k-nearest neighbour approach, which was proven to be competitive across a variety of tasks despite its simplicity [12]. Wide and Deep consists of training jointly wide linear models and deep neural networks to combine the benefits of memorization and generalization for recommender systems. Finally, Factorization Machines (FM) is a classic competitive baselines when dealing with cold-start recommendations [30] and explicitly model first and second order interactions of feature and click embeddings. We also experimented with CLCRec [46], and its extension EQUAL [45]. Finally, we experimented with CVAR [49], however the code repository of [49] did not scale to the size of our datasets.

Comparison with baselines. For fair comparison, we adopt the same featurization strategies reported in [6, 46] to produce the feature matrix $\mathbf{F}$, that is, the text features are embedded through tf-idf with 1000 as vocabulary size and the categorical features through multi-hot encoding. For the non-negative decreasing function h used in the regularization term $\mathbf{D}^{\mathrm{R}}$ of equation (6), we use a step linear decreasing function. We take the 10^{th} percentile p of the distribution of the number of ratings per item $r^{(i)} = \sum_i X_{.i}$ and the linear function is written $h\left(r^{(i)}\right) = k\left(p - r^{(i)}\right)$ if $r^{(i)} \leq p$, $h\left(r^{(i)}\right) = 0$ if $r^{(i)} \geq p$, with $k = \beta/p$. We keep $f_\phi^E = I$ for the embedding model and for the alignment model f_ζ^A we used equation (5), and optimize for the first and second order scalar terms. For the backbone model f_θ^B, we used equation (9) which is the fastest to run.

Table 3 shows the lifts obtained by MARec on the MovieLens Hetrec, Amazon Video Games, MovieLens10M, and Netflix datasets, all with significant margins. Hyper-parameter optimization was performed on validation sets for all models via grid search. Regarding MARec, the hyper-parameters obtained on the MovieLens Hetrec validation sets are $\delta = 20$ (equation 5), $\lambda_1 = 1$ (equation 9), $\alpha = 1$ (equation 6), $\beta = 100$ (term $\mathbf{D}^{\mathrm{R}}$ in equation 6). For the Amazon Video Games dataset, the hyperparameters obtained on the validation sets are $\delta = 50$, $\lambda_1 = 1$, $\alpha = 10$, $\beta = 100$. We note that learning the first and second order coefficients in the alignment function of (5) did not lead to improvements, which we relate to the high sparsity observed in the data (click percentage of 0.06%, see Table 1). For the MovieLens10M dataset, the hyperparameters obtained on the validation set are $\delta = 50$, $\lambda_1 = 700$, $\alpha = 1$, $\beta = 60$.

	MovieLens Hetrec			
	hr@10	ndcg@10	hr@25	ndcg@25
NFC	0.1904	0.2076	0.1748	0.1866
ItemKNNCF	0.1175	0.1335	0.1130	0.1214
Wide and Deep	0.1555	0.1762	0.1479	0.1588
FM	0.1790	0.1946	0.1808	0.1842
CLCRec	0.0815	0.0763	0.0909	0.0848
EQUAL	0.1310	0.1470	0.1124	0.1252
MARec (Eq 9)	**0.2928**	**0.3071**	**0.2717**	**0.2826**
Lift	+53.8%*	+47.9%*	+49.7%*	+51.4%*
	Amazon Video Games			
	hr@10	ndcg@10	hr@25	ndcg@25
NFC	0.1231	0.0785	0.2099	0.1040
ItemKNNCF	0.1187	0.0786	0.1842	0.0980
Wide and Deep	0.0103	0.0060	0.0200	0.0088
FM	0.0080	0.0134	0.0129	0.0302
CLCRec	0.1031	0.0676	0.1804	0.0903
EQUAL	0.1171	0.0778	0.1957	0.1007
MARec (Eq 9)	**0.1568**	**0.1047**	**0.2433**	**0.1305**
Lift	+27.4%*	+33.2%*	+15.9%*	+25.5%*
	MovieLens10M			
	hr@10	ndcg@10	hr@25	ndcg@25
NFC	0.0322	0.0379	0.0342	0.0366
ItemKNNCF	0.0961	0.0865	0.1525	0.1054
Wide and Deep	0.0576	0.0571	0.0864	0.0654
FM	0.0718	0.0634	0.1064	0.0737
CLCRec	0.0752	0.0443	0.1102	0.0851
EQUAL	0.0864	0.0626	0.1309	0.1010
MARec (Eq 9)	**0.1094**	**0.0938**	**0.1956**	**0.1236**
Lift	+13.8%*	+8.4%*	+28.3%*	+17.3%*
	Netflix			
	hr@10	ndcg@10	hr@25	ndcg@25
NFC	0.1087	0.1098	0.1278	0.1114
ItemKNNCF	0.1081	0.1115	0.1249	0.1127
Wide and Deep	0.0188	0.0146	0.0250	0.0180
FM	0.0586	0.0483	0.0990	0.0651
CLCRec	0.0934	0.0987	0.1123	0.1074
EQUAL	0.1013	0.1027	0.1252	0.1109
MARec (Eq 9)	**0.1369**	**0.1412**	**0.1574**	**0.1411**
Lift	+25.9%*	+26.6%*	+23.2%*	+25.2%*

Table 3: Baseline comparison of MARec across benchmarking datasets on cold-start recommendation. Top performing and second best scores are respectively in bold and underlined. Lift is reported between MARec and best performing baseline and * indicates statistical significance.

[3] https://github.com/cesarebernardis/NeuralFeatureCombiner

Finally, for the Netflix dataset, the hyperparameters obtained on the validation set are $\delta = 100$, $\lambda_1 = 500$, $\alpha = 1$, $\beta = 100$.

4.4 Ablation Studies

Here we focus on MovieLens10M dataset although similar observation was made on other datasets as well and those results are not presented for sake of brevity.

Discussion on the encoding and embeddings functions. We have showed that our proposed approach can outperform competitive baselines on a variety of cold-start dataset splits. This can be further improved by utilising better embeddings and leveraging more features. As a result, we are first interested in the recommendation quality we get without the fusion term, by only considering item metadata.

	MovieLens10M			
	hr@10	hr@25	hr@50	hr@100
ItemKNN	0.0071	0.0137	0.0241	0.0449
MARec (Eq 5, 1st order)	0.0326	0.0484	0.0757	0.1235
MARec (Eq 5)	0.0329	0.0487	0.0776	0.1302
MARec (Eq 5, Falcon7B)	<u>0.0373</u>	<u>0.0612</u>	<u>0.1007</u>	<u>0.1545</u>
MARec (Eq 5, Falcon7B (all))	0.0344	0.0541	0.0905	0.1448
MARec (Siamese, Falcon7B (all))	0.0357	0.0537	0.0820	0.1280
MARec (Eq 5, ST)	**0.0378**	**0.0626**	**0.1028**	**0.1602**

Table 4: Ablation study on the encoding and embeddings functions for metadata-based recommendations on the MovieLens10M dataset split. Top performing and second best scores are respectively in bold and underlined.

Table 4 presents the results of different encoding, and embeddings strategies to get to metadata-based recommendations. The first step we tried was to replace the text tf-idf embeddings with pretrained Large Language Models (LLMs) embeddings: we experimented with Falcon-7B [4] and Sentence Transformers (ST) [33] embeddings. We also experimented with learning the first order coefficients $\mu_i, i \in \{1, \cdots, N\}$ vs learning first and second order coefficients μ_i, μ_{ij}, with $i, j \in \{1, \cdots, N\}$, $j > i$ of equation (5). First, we notice that LLM pre-trained embeddings largely outperformed the tf-idf embeddings for hr@k and ndcg@k, for instance ST embeddings led to an increase of +22.8% for hr@100 with respect to tf-idf embeddings. Second, we notice that learning scalar weights for second order terms, as per equation (5), brings a clear gain in ranking metrics. The second and third rows of Table 4 show a consistent benefit over ranking metrics and a gain of 5.4% for hr@100. This is because the multiplication of cosine similarities enable to model nonlinear interactions between sub-feature spaces.

Finally, we also report results on converting the categorical features (actors, directors, genres) into numeric representations with Falcon7B, indicated as "Falcon7B (all)" in Table 4. This led to a small decrease in performance over using multi-hot encoded features, e.g., -6.8% for hr@100. However this helped when training the Siamese network, and the best results on cold-start splits were reported using Falcon embeddings across all features. For the Siamese network, we used 1 projection layer followed by 1 cross attention layer between text and categorical features and 2 dense layers. We noticed that feeding categorical features to the Siamese network was leading the network to learn id associations between multi-hot encoded features and items, instead of semantic ones. The results we report here show that the Siamese network fall short of the best reported results, e.g., -6.6% for recall@10.

Fusion performance, and comparison of backbone models. The previously presented results give an indication of the performance of the fusion approaches (equations 7 and 9), that we present in Table 5.

	MovieLens10M			
	hr@10	hr@25	hr@50	hr@100
MARec (Eq 9, ST)	**0.1385**	**0.2240**	0.3286	0.4692
MARec (Eq 9, BoW)	0.1094	0.1956	0.2866	0.4203
MARec (Eq 9, Siamese)	0.0910	0.1601	0.2457	0.4109
MARec (Eq 7, ST)	0.0779	0.2037	**0.3831**	**0.5072**
MARec (Eq 9, ST)	**0.1385**	**0.2240**	0.3286	0.4692
MARec (Eq 10, ST)	<u>0.1282</u>	<u>0.2204</u>	<u>0.3421</u>	<u>0.4789</u>

Table 5: Ablation study on the MovieLens10M dataset for difference fusion strategies, and for different backbone models. Top performing and second best scores are respectively in bold and underlined.

First, the Siamese approach which was trailing best results on the metadata-based learning task is also trailing after fusion. It falls behind the alignment function leveraging first and second order cosine similarities, as per equation (5), by 2.3% on recall@100. As future work, we will explore improving the joint representation learning of item metadata with the Siamese network, by e.g., including external data to further help with the adaptation of the pre-trained embeddings. Second, we report the performance of our network with the 3 backbone models of equations (7), (9), and (10). The multi-VAE network was trained with one encoding and one decoding layer similarly to [24] and after hyper-parameter turning, we trained the network over 20 epochs, with a embedding dimension of 200, and D^{A} in equation (6) chosen such that $\beta = 50$. Our observation is that equation (7) struggles with ranking metrics for low values of k, falling -44.8% and -9.1% behind (9) for hr@10 and hr@25, while outperforming (9) and (10) for high values of k, with a gain of +16.6% for hr@50 and +8.1% for hr@100. We interpret that this is caused by the needs for deeper representations with more than one hidden layer, combined with the inaccuracy of the sampling Gaussian prior. We note that this difference in performance between the two backbone models was also appreciated in [41], which focused on warm conditions, and where a late fusion was proposed to ensemble a multi-VAE branch and a neural EASE branch, by simply multiplying the softmax probabilities obtained. As future work, we may experiment with similar ensembling techniques to bring the best out of the backbone models, see e.g. [20].

Adding images, and tags. We explore the incorporation of additional metadata into our framework. Table 6 presents the results when we incorporated image embeddings and item tags to the metadata signals.

	MovieLens10M			
	hr@10	hr@25	hr@50	hr@100
MAREC (Eq 9, ST)	0.1385	0.2240	0.3286	0.4692
MAREC (Eq 9, ST + images)	0.1401	0.2267	0.3315	0.4721
MAREC (Eq 9, ST + images, tags)	0.2194	0.3540	0.4808	0.6168

Table 6: Results for the cold-start MovieLens10M dataset split on introducing additional metadata to our framework.

The MovieLens10M images were pulled from the Kaggle poster repository[4]. We used the a pretrained CLIP model (ViT-H/14 - LAION-2B[5]) to get image embeddings. We relied on this model as we also tried to generate keywords and captions from these images by leveraging the BLIP2-2.7b architecture but this did not lead to improvements in ranking metrics. On the contrary, Table 6 shows clear benefits in adding the image embeddings to the framework, with gains ranging from 0.5% to 1.4% on the recall@k metrics. Finally, we also made use of the MovieLens tags provided as part of the MovieLens dataset and we simply encoded them through multi-hot encoding, after merging similar tags together, which led to a vocabulary of 2504 words. This is an informative feature, e.g., the movie "Dumb and Dumber" would come with the following tags: "Jim Carrey", "Jeff Daniel", "stupid", "comedy", "infantil", "hilarious".

The inclusion of such tags led to a very significant additional improvement in ranking metrics, e.g. +30.7% for hr@100. We also get to an impressive cold-start performance of 0.6168 for hr@100, which after checking is only 15.1% behind the hr@100 we get on the warm split. In general, the inclusion of semantic features, i.e. LLM embeddings, image embeddings, and tags metadata, have led to a gain of 105.5% on hr@10 and +46.7% on hr@100 with respect to training the same model with BoW representation. Under the MovieLens set-up, tags are coming from user entries, however, we believe the process of tagging items could be repeated by a ML system utilising the world knowledge hence removing the needs for an user input. We leave this as future work.

Results for transitioning from cold to warm conditions. Finally, we report the capability of the proposed approach to provide relevant recommendations for situations in which there is a mix of cold and warm items.

Table 7 presents the results in the 3 test scenarios: warm items only, cold and warm items, and cold items only, against the 2 best performing approaches reported in [46]. We have seen that MAREC was largely beating SOTA results on cold-start splits and here we confirm its efficacy for warm and mixed situations. As for the hyperparameters, we get $\lambda_1 = 700$, $\alpha = 1$, $\beta = 0.5$, for the mixed split and $\lambda_1 = 500$, $\alpha = 1$, $\beta = 0$, for the warm split. For the mixed and warm situations, it also beats the baselines, with CLCRec and EQUAL following more closely. We repeat that our approach transitions to the backbone model in warm set ups, which can be selected from top performing collaborative filtering algorithms.

A note on time complexity. We note that the training process can be done either in sequential steps by first learning the best

[4]https://www.kaggle.com/datasets/ghrzarea/movielens-20m-posters-for-machine-learning?select=MLP-20M
[5]laion/CLIP-ViT-H-14-laion2B-s32B-b79K

	MovieLens10M					
	Warm		Cold		All	
	hr@10	ncdg@10	hr@10	ndcg@10	hr@10	ndcg@10
NFC	0.0945	0.0736	0.0322	0.0379	0.0833	0.0767
ItemKNNCF	0.0499	0.0476	<u>0.0962</u>	<u>0.0865</u>	0.0509	0.0565
Wide and Deep	0.1204	0.0907	0.0576	0.0571	0.0879	0.0752
FM	0.1461	0.1206	0.0718	0.0634	0.1012	0.0947
CLCRec	0.3157	0.2364	0.0752	0.0443	0.2335	0.1979
EQUAL	<u>0.3368</u>	<u>0.2471</u>	0.0864	0.0626	<u>0.2427</u>	<u>0.2056</u>
MAREC (Eq 5, ST)	0.0558	0.0494	0.0378	0.0384	0.0618	0.0662
MAREC (Eq 9, ST)	**0.3513**	**0.2890**	**0.1385**	**0.1150**	**0.2443**	**0.2283**

Table 7: Results on the MovieLens10M dataset under different evaluation scenarios with warm, mixed, and cold items. Top performing and second best scores are respectively in bold and underlined.

metadata representations (equations 3, 5) and then training the backbone and fusion part of the network (equations 7, 9, 11), or in an end-to-end fashion. The sequential implementation enables not to pass all the metadata feature matrix **F** at test time but rather the user specific generated embeddings. We report the training times of the approaches: on MovieLens10M, 0.43 s for ItemKNNCF, 5.3e3 sec for Wide and Deep, 1.3e3 s for FM, 1.3e4 s for NFC, 6.3e4 s for EQUAL, 230 s for (7), and 34 s for (9); and on Amazon Video Games, 2 s for ItemKNNCF, 190 s for Wide and Deep, 27 s for FM, 8.2e3 s for NFC, 1.8e3 s for EQUAL, 440 s for equation (7), and 92 s for equation (9). The gain in training time over the second best performing approach is of one order of magnitude for the VAE backbone and two orders of magnitude for the EASE backbone. We note that the matrix implementation of (9) will suffer the curse of dimensionality in the user space, however equation (7) scales well as it trains across batches of users.

4.5 Results for Warm Splits

We also put a focus on warm dataset splits to evaluate the merit of our proposed optimization strategy of equation (10), and its closed form solution presented in equation (11). As for the weight matrices we use $w_0 = 1$ for positive labels and tunable parameter w_1 for negative labels. Hence the 3 parameters to optimise are w_1, λ_1, and γ_1. For the weight matrices we use $w_0 = 1$ for positive labels and tunable parameter w_1 for negative labels. Hence the 3 parameters to optimise are w_1, λ_1, and γ_1. Hyperparameter optimization is done via grid search on the validation set optimising for both ndcg@10 and recall@10, where the grids are defined with parameters such as $w_1 \in \{0, 0.5, ..., 1\}$, $\lambda_1 \in \{0, 100, ..., 2000\}$, and $\gamma_1 \in \{0, 100, ..., 2000\}$. We get $w_1 = 0.25$, $\lambda_1 = 300$, $\gamma_1 = 800$ for MovieLens1M, $w_1 = 0.05$, $\lambda_1 = 300$, $\gamma_1 = 100$ for Pinterest. We implement our approach with a batch array job with on-demand AWS Fargate resources and 1 vCPU of 8 GB of memory per job, which runs in 1 min for the MovieLens dataset and 7 min for the Pinterest dataset.

In Table 8, we report the competitive baselines already reported in [35], to which we add the EASE model [38] and our optimization strategy (11). The baselines are the neural matrix factorization (NMF) approach of [17], the implicit alternating least squares (iALS) collaborative filtering method [36], the original SLIM model [28], and the matrix factorization (MF) implementation of [35] which

	MovieLens1M		Pinterest	
	hr@10	ncdg@10	hr@10	ndcg@10
Popularity	0.4535	0.2543	0.2740	0.1409
iALS	0.711	0.4383	0.8762	0.5590
NeuMF	0.7093	0.4039	<u>0.8777</u>	0.5576
SLIM	0.7162	0.4468	0.8679	0.5601
MF	<u>0.7294</u>	<u>0.4523</u>	**0.8895**	**0.5794**
EASE	0.7184	0.4494	0.8634	0.5616
MARec (Eq 10)	**0.7301**	**0.4560**	0.8748	<u>0.5683</u>
Lift	+0.1%	+0.8%*	-1.5%	-1.9%*

Table 8: Baseline comparison of MARec on warm-start benchmarking datasets. Top performing and second best scores are respectively in bold and underlined. Lift is reported between MARec and best performing baseline and * indicates statistical significance.

is the reported SOTA result on the 2 datasets. It is interesting to note that weighted least squares beats matrix factorization for the least sparse dataset (average margin +0.5%) and fails to beat it for the sparse dataset (average margin -1.7%). Another remark is that we consistently beat SLIM (+2.0%, +1.1%), and EASE (+1.5%, +1.3%), meaning that our closed-form weighted least squares should always be considered as strong baseline.

5 CONCLUSION

We introduced MARec, a novel algorithm to enhance cold start recommendations by integrating semantic information. This was achieved by combining similarities derived from user click actions and metadata features using a regularization term, aligning cold-start items with the collaborative filtering space. Experimental results demonstrated that MARec significantly outperformed reported SOTA results on cold-start datasets with varying sparsity and metadata characteristics, showing improvements ranging from +8.4% to +53.8% on hr@k and ndcg@k metrics while being competitive on warm-start recommendation datasets. Moreover, MARec achieved these results while being one order of magnitude faster to train compared to the second-best performing baseline. The inclusion and utilization of semantic features led to substantial gains ranging from +46.8% to +105.5% compared to Bag-of-Words (BoW) representations. Future work will focus on refining joint metadata embeddings by incorporating external data sources, exploring generative models for informative keyword generation, and investigating more sophisticated end-to-end training and merging mechanisms within the framework.

REFERENCES

[1] [n. d.]. IMDb Non-Commercial Datasets. https://developer.imdb.com/non-commercial-datasets/.
[2] [n. d.]. MovieLens 10M Dataset. https://grouplens.org/datasets/movielens/10m/.
[3] [n. d.]. Netflix Prize data. https://www.kaggle.com/datasets/netflix-inc/netflix-prize-data.
[4] Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. The Falcon Series of Open Language Models. arXiv:2311.16867 [cs.CL]
[5] Bing Bai and Yushun Fan. 2017. Incorporating field-aware deep embedding networks and gradient boosting decision trees for music recommendation. In *The 11th ACM International Conference on Web Search and Data Mining (WSDM). ACM, London, England*, Vol. 7.
[6] Cesare Bernardis and Paolo Cremonesi. 2021. NFC: a deep and hybrid item-based model for item cold-start recommendation. *User Modeling and User-Adapted Interaction* (2021), 1–34.
[7] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*. PMLR, 2206–2240.
[8] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1993. Signature Verification Using a "Siamese" Time Delay Neural Network. In *Proceedings of the 6th International Conference on Neural Information Processing Systems* (Denver, Colorado) *(NIPS'93)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 737–744.
[9] Hung-Hsuan Chen and Pu Chen. 2019. Differentiating regularization weights–A simple mechanism to alleviate cold start in recommender systems. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 13, 1 (2019), 1–22.
[10] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.
[11] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. 191–198.
[12] 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*. 101–109.
[13] Mukund Deshpande and George Karypis. 2004. Item-Based Top-N Recommendation Algorithms. *ACM Trans. Inf. Syst.* 22, 1 (jan 2004), 143–177. https://doi.org/10.1145/963770.963776
[14] Xiaoyu Du, Xiang Wang, Xiangnan He, Zechao Li, Jinhui Tang, and Tat-Seng Chua. 2020. How to learn item representation for cold-start multimedia recommendation?. In *Proceedings of the 28th ACM International Conference on Multimedia*. 3469–3477.
[15] Zeno Gantner, Lucas Drumond, Christoph Freudenthaler, Steffen Rendle, and Lars Schmidt-Thieme. 2010. Learning attribute-to-feature mappings for cold-start recommendations. In *2010 IEEE international conference on data mining*. IEEE, 176–185.
[16] Xue Geng, Hanwang Zhang, Jingwen Bian, and Tat-Seng Chua. 2015. Learning Image and User Features for Recommendation in Social Networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*. 4274–4282. https://doi.org/10.1109/ICCV.2015.486
[17] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*. 173–182.
[18] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *2008 Eighth IEEE International Conference on Data Mining*. 263–272. https://doi.org/10.1109/ICDM.2008.22
[19] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2019. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172* (2019).
[20] Daeryong Kim and Bongwon Suh. 2019. Enhancing VAEs for collaborative filtering: flexible priors & gating mechanisms. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 403–407.
[21] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 426–434.
[22] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.
[23] Xuan Nhat Lam, Thuc Vu, Trong Duc Le, and Anh Duc Duong. 2008. Addressing cold-start problem in recommendation systems. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*. 208–211.
[24] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational autoencoders for collaborative filtering. In *Proceedings of the 2018 world wide web conference*. 689–698.
[25] Alexander Long, Wei Yin, Thalaiyasingam Ajanthan, Vu Nguyen, Pulak Purkait, Ravi Garg, Alan Blair, Chunhua Shen, and Anton van den Hengel. 2022. Retrieval augmented classification for long-tail visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 6959–6969.
[26] Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. 2011. Content-based recommender systems: State of the art and trends. *Recommender systems handbook* (2011), 73–105.

[27] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. 43–52.

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

[29] Xia Ning and George Karypis. 2012. Sparse linear methods with side information for top-n recommendations. In *Proceedings of the sixth ACM conference on Recommender systems*. 155–162.

[30] Feiyang Pan, Shuokai Li, Xiang Ao, Pingzhong Tang, and Qing He. 2019. Warm up cold-start advertisements: Improving ctr predictions via learning to learn id embeddings. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 695–704.

[31] Alexandrin Popescul, Lyle H Ungar, David M Pennock, and Steve Lawrence. 2013. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. *arXiv preprint arXiv:1301.2303* (2013).

[32] Zhen Qin, Le Yan, Honglei Zhuang, Yi Tay, Rama Kumar Pasumarthi, Xuanhui Wang, Mike Bendersky, and Marc Najork. 2021. Are Neural Rankers still Outperformed by Gradient Boosted Decision Trees? (2021).

[33] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. https://arxiv.org/abs/1908.10084

[34] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International conference on data mining*. IEEE, 995–1000.

[35] Steffen Rendle, Walid Krichene, Li Zhang, and John Anderson. 2020. Neural collaborative filtering vs. matrix factorization revisited. In *Fourteenth ACM Conference on Recommender Systems*. 240–248.

[36] Steffen Rendle, Walid Krichene, Li Zhang, and Yehuda Koren. 2022. Revisiting the performance of ials on item recommendation benchmarks. In *Proceedings of the 16th ACM Conference on Recommender Systems*. 427–435.

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

[38] Harald Steck. 2019. Embarrassingly shallow autoencoders for sparse data. In *The World Wide Web Conference*. 3251–3257.

[39] Harald Steck. 2020. Autoencoders that don't overfit towards the identity. *Advances in Neural Information Processing Systems* 33 (2020), 19598–19608.

[40] Harald Steck, Maria Dimakopoulou, Nickolai Riabov, and Tony Jebara. 2020. ADMM SLIM: Sparse Recommendations for Many Users. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 555–563.

[41] Vojtěch Vančura and Pavel Kordík. 2021. Deep variational autoencoder with shallow parallel path for top-N recommendation (VASP). In *Artificial Neural Networks and Machine Learning–ICANN 2021: 30th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 14–17, 2021, Proceedings, Part V 30*. Springer, 138–149.

[42] Manasi Vartak, Arvind Thiagarajan, Conrado Miranda, Jeshua Bratman, and Hugo Larochelle. 2017. A meta-learning perspective on cold-start recommendations for items. *Advances in neural information processing systems* 30 (2017).

[43] Maksims Volkovs, Guangwei Yu, and Tomi Poutanen. 2017. Dropoutnet: Addressing cold start in recommender systems. *Advances in neural information processing systems* 30 (2017).

[44] Tongzhou Wang and Phillip Isola. 2020. Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere. In *Proceedings of the 37th International Conference on Machine Learning (ICML'20)*. JMLR.org, Article 921, 11 pages.

[45] Wenjie Wang, Xinyu Lin, Liuhui Wang, Fuli Feng, Yinwei Wei, and Tat-Seng Chua. 2023. Equivariant Learning for Out-of-Distribution Cold-start Recommendation. In *Proceedings of the 31st ACM International Conference on Multimedia*. 903–914.

[46] Yinwei Wei, Xiang Wang, Li Qi, Liqiang Nie, Yan Li, Xuanqing Li, and Tat-Seng Chua. 2021. Contrastive Learning for Cold-start Recommendation. In *Proceedings of the 29th ACM International Conference on Multimedia*.

[47] Xiaoxiao Xu, Chen Yang, Qian Yu, Zhiwei Fang, Jiaxing Wang, Chaosheng Fan, Yang He, Changping Peng, Zhangang Lin, and Jingping Shao. 2022. Alleviating Cold-start Problem in CTR Prediction with A Variational Embedding Learning Framework. In *Proceedings of the ACM Web Conference 2022*. 27–35.

[48] Yan Zhang, Hongzhi Yin, Zi Huang, Xingzhong Du, Guowu Yang, and Defu Lian. 2018. Discrete deep learning for fast content-aware recommendation. In *Proceedings of the eleventh ACM international conference on web search and data mining*. 717–726.

[49] Xu Zhao, Yi Ren, Ying Du, Shenzheng Zhang, and Nian Wang. 2022. Improving item cold-start recommendation via model-agnostic conditional variational autoencoder. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2595–2600.

[50] Ziwei Zhu, Yun He, Xing Zhao, Yin Zhang, Jianling Wang, and James Caverlee. 2021. Popularity-Opportunity Bias in Collaborative Filtering. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (Virtual Event, Israel) *(WSDM '21)*. Association for Computing Machinery, New York, NY, USA, 85–93. https://doi.org/10.1145/3437963.3441820