# Transfer Learning for Better Cold-Start Recommendation Using Multiple Domains

Rasmus Krusaa
Department of Computer Science
rkrusa14@student.aau.dk

Christian Kloster
Department of Computer Science
cklost15@student.aau.dk

Victor Kamp
Department of Computer Science
vkamp14@student.aau.dk

## ABSTRACT

Much work has been devoted to utilize data from other domains than the domain we want to recommend in. Typically, we want to recommend in the item domain. Two very popular types domain models are the social-aware systems and questionnaire-based approaches. Social data has shown to reflect users preferences and questionnaires in its essence provide additional data about users preferences. One way of making use of data in multiple domains is to utilize transfer learning. The idea behind transfer learning is, as implied by the name, to learn how to transfer knowledge between domains.

In this work, we propose a *Questionnaire-Based Efficient Adaptive Transfer Neural Network*, QEATNN, which transfers knowledge from the social and questionnaire domain into the item domain to alleviate the cold-start user problem. We get our questionnaire by employing a state-of-the-art questionnaire-based approach, *Local Representative-Based MF*(LRMF). QEATNN automatically learns how to transfer knowledge between domains by using attention modules. Our model is jointly optimized by computing a loss for each of the domains, i.e. item, social and questionnaire domain.

Extensive experiments on three publicly available datasets have been conducted to validate QEATNN's performance in different settings. We measure performance with the widely used metrics *NDCG*, *Precision* and *Recall*. The results show that we perform comparably with state-of-the-art baselines. In some settings QEATNN outperforms the baselines. One of these settings is the cold-start setting, where we outperform our base model, EATNN, on shorter lists.

## Keywords

Recommender Systems, Cold-Start User Recommendation, Social Network, Questionnaires, Decision Trees, Transfer Learning

## 1. INTRODUCTION

Being in an era where online websites are part of many daily lives recommender systems are essential. Recommender systems, in its essence, aim to find good choices in an overload of information. Recently, many of these online websites utilizing recommender systems have become popular social platforms, where users can follow or befriend other users and interact with items. The social connections of these websites can be used for better personalized recommendations, as they to some degree reflect users preferences.

Collaborative Filtering (CF)[11, 13, 14, 20] approaches has proven to perform well, because of their ability to make use of historical interactions in order to obtain good recommendations. However, they struggle whenever the amount of historical interactions is limited. In other words, if the interaction matrix is sparse and users are *cold*. Cold users mean users who have few to no historical interactions. Previous work[4, 21, 16, 22, 19] has addressed this problem by utilizing data from different domains, such as social networks or questionnaires, with the goal of getting better recommendations. Where the websites typically provide a social network, data from a questionnaire is easily obtained by simply interviewing users with a given questionnaire.

Intuitively, if we have data for interactions, social connections and answers to a questionnaire, cold users' preferences are better described by a combination of data from these domains than solely by their few interactions or relying on just one of the domains. Therefore, models which are able to utilize data from all three domains (interactions, social networks and questionnaires) can tackle the data-sparsity problem and make better recommendations.

Transfer learning, which is used in many fields such as text classification[6] and computer vision[17], has become popular for transferring knowledge between domains in recommender systems[8]. As the name implies, the concept behind transfer learning is to learn how to transfer knowledge from some domain into another domain. This way, we can make use of data from interactions, social networks and answered questionnaire. Where most models using transfer learning transfers the same amount of knowledge between domains, we will transfer a personalized amount for each user. This is because 1) the information in the three domains (interactions, social network and answered questionnaires) varies from user to user and 2) users' preferences might be reflected differently in the three domains.

Based on the above-mentioned, we pose the following problem definition:

> *How can interactions, social networks and answered questionnaires alleviate the cold-start user problem?*

To answer this problem definition, we propose an extension of *Efficient Adaptive Transfer Neural Network*[5], EATNN, in which we make use of three domains: interactions, social networks and answered questionnaires. We name our approach *Questionnaire-Based EATNN* (QEATNN). The idea behind QEATNN is to introduce attention mechanisms[1, 3] which automatically learn the amount of knowledge for each user that has to be shared between the domains. In QEATNN we assume, that we already have a questionnaire with which we can interview users. However, obtaining questionnaires is in itself a problem which much work has been devoted to[15, 19, 23]. Therefore, an obvious extension of QEATNN could be one with a more integrated learning of the questionnaire.

The main contributions of this paper can be summarized into:

1. We propose a model, *QEATNN*, utilizing three domains of data: interactions, social and questionnaire. The model automatically learns the interplay of the three domains in attention modules for each user.
2. We evaluate QEATNN's performance on two publicly available datasets and compare with our base model EATNN as well as other state-of-the-art social-aware and questionnaire-based approaches.
3. We perform extensive experiments in different settings and also here compare with our base model EATNN and other state-of-the-art social-aware and questionnaire-based approaches.

The paper is structured such that in section 2 we will present related work. In section 3 we will describe notations and preliminaries. In section 4, QEATNN will formally be described as well as EATNN and how we obtain a questionnaire. In section 5 we will conduct experiments. Finally, in section 6 we will conclude our work and suggest further work.

| | |
|---|---|
| $\mathbf{U}$ | set of users |
| $\mathbf{B}$ | batch of users |
| $\mathbf{V}$ | set of items |
| $\mathbf{R}$ | user-item interaction matrix |
| $\mathcal{R}$ | the set of non-zero user-item pairs |
| $\mathbf{G}$ | user-user social matrix |
| $\mathcal{G}$ | the set of non-zero user-user pairs |
| $\mathbf{Q}$ | user-question questionnaire matrix |
| $\mathcal{Q}$ | the set of non-zero user-question pairs |
| $\mathbf{u}^I$ | item specific latent vector of user $u$ |
| $\mathbf{u}^S$ | social specific latent vector of user $u$ |
| $\mathbf{u}^Q$ | questionnaire specific latent vector of user $u$ |
| $\mathbf{u}^C$ | common latent vector of user $u$ |
| $\mathbf{p}_u^I$ | latent vector of user $u$ in item domain after transferring |
| $\mathbf{p}_u^S$ | latent vector of user $u$ in social domain after transferring |
| $\mathbf{p}_u^Q$ | latent vector of user $u$ in questionnaire domain after transferring |
| $\mathbf{q}_v$ | latent vector of item $v$ |
| $\mathbf{g}_t$ | latent vector of user $t$ as a friend |
| $\alpha_{(I,u)}$ | the weight of $\mathbf{u}^I$ in item domain |
| $\alpha_{(C,u)}$ | the weight of $\mathbf{u}^C$ in item domain |
| $\beta_{(S,u)}$ | the weight of $\mathbf{u}^S$ in social domain |
| $\beta_{(C,u)}$ | the weight of $\mathbf{u}^C$ in social domain |
| $\gamma_{(Q,u)}$ | the weight of $\mathbf{u}^Q$ in questionnaire domain |
| $\gamma_{(C,u)}$ | the weight of $\mathbf{u}^C$ in questionnaire domain |
| $d$ | number of latent factors |
| $\Theta$ | our models' parameters |

**Table 1: Notations and description.**

## 2. RELATED WORK

In this section we will review related work by looking at two broad types of recommender systems: social-aware and questionnaire-based recommender systems.

### 2.1 Social-Aware Recommendation

Much work has been devoted to social-aware recommendation[22, 4, 21, 5]. Social-aware recommender systems works by leveraging data from a social domain to obtain better recommendations in, for example, the item domain. This can be done in multiple ways. In SBPR[22], Zhao et al. assume that users are more likely to interact with items their friends (from the social network) have interacted with than items they have not interacted with themselves and extended BPR[20] with this assumption. Another way is to use transfer learning, such that we transfer knowledge from the social domain into another domain in which we want to be able to recommend. This is exactly what Xiao et al. did in TranSIV[21] where they used transfer learning to combine the social domain with the item domain. Finally, deep learning has been widely adopted in many fields, and, naturally, work has been devoted to use deep learning for social-aware recommendation[5, 4]. In EATNN[5], our base model, a network is constructed for capturing the interplay of both the social domain and item domain. In SAMN[4], Chen et al. propose a model which considers both user specific and friend specific aspects.

### 2.2 Questionnaire-Based Approaches

Questionnaire-based approaches has proven to be great way to mitigate the cold-start problem[15, 19, 23]. The biggest advantage of such an approach is its' ability to generate additional data about a user. This is typically an opinion of an item, such as an explicit rating or indication of consumption, but it can also be more indirect data, such as an opinion of an item attribute (for example opinion about an actor).

The most common approaches are based on the idea that users with the same answers to a set of questions are similar in regards to preferences[15, 19, 23]. Some of these models, utilize a tree-based structure to model the interview process and group appropriate users.

Questionnaires can theoretically be used in any model that can leverage user information in any way, as long as it can be formulated as the, preferably simple, answer to a question. This flexibility also speaks to the potential of questionnaire-based models.

# 3. PRELIMINARIES

In this section we will introduce key notations used in this work.

Table 1 shows notations and a short corresponding description of key concepts used in the following sections. If we have $N$ users and $M$ items, for our interaction matrix we get $\mathbf{R} \in \mathbb{R}^{N \times M}$, and we denote user $u$'s interaction on item $i$ as $R_{ui}$. Furthermore, for our social matrix we get $\mathbf{G} \in \mathbb{R}^{N \times N}$ and we denote the friendship between user $u$ and user $v$ as $G_{uv} \in \{0, 1\}$ (1 if $v$ is a friend of $u$, 0 if not). For the questionnaire matrix we get $\mathbf{Q} \in \mathbb{R}^{N \times T}$, where $T$ denotes number of questions we use in the questionnaire. Since users might be asked different questions, we denote the questions we ask user $u$ as $Q_u \in \mathbb{R}^{T}$. To obtain answers for the questionnaire we make look-ups in the interaction matrix.

User $u$ is represented by four latent vectors: $\mathbf{u}^I$, $\mathbf{u}^S$, $\mathbf{u}^Q$, $\mathbf{u}^C$. Item $i$ is represented by a latent vector $\mathbf{q}_v$ and user $t$ as friend is represented by latent vector $\mathbf{g}_t$. $\mathbf{p}_u^I$, $\mathbf{p}_u^S$ and $\mathbf{p}_u^Q$ denotes user $u$ representation after transferring in the item, social and questionnaire domain respectively. $\alpha$ are item domain specific weights, $\beta$ are social domain specific weights and $\gamma$ are questionnaire domain specific weights. This will more detailedly be described in section 4.

# 4. PROPOSED MODEL

In this section, we first present an overview of our proposed model. Then, we describe the mechanism for selecting questions to the questionnaire followed by a description of the predecessor (i.e. the model, EATNN[5], which we extend) of our proposed model. Finally, we describe our proposed model in more detail as well as how we learn the model.

## 4.1 Model Overview

The goal of our work is to improve the recommendations in the item domain by also utilizing knowledge from the social domain and answered questionnaires. Figure 1 depicts a high-level abstraction of our proposed model, which utilizes knowledge from the item domain, social domain and answered questionnaires. Looking at the figure we can see:

1. Users, items, friends and questions are converted to dense vector representations through embeddings. Specifically, users are converted to four dense vector representations: $\mathbf{u}^I$ representing user $u$'s preferences in the item domain, $\mathbf{u}^S$ representing user $u$'s preferences in the social domain, $\mathbf{u}^Q$ representing user $u$'s preferences based on his answered questionnaire and $\mathbf{u}^C$ representing knowledge shared between the three domains: item domain, social domain and answered questionnaires.

2. Transferring knowledge between the three domains happens in the adaptive modules coloured with orange. These modules are designed such that they automatically learn relationships between the domain in question and shared representations, i.e. relationships between $\mathbf{u}^x$ and $\mathbf{u}^C$ where $x \in \{I, S, Q\}$.

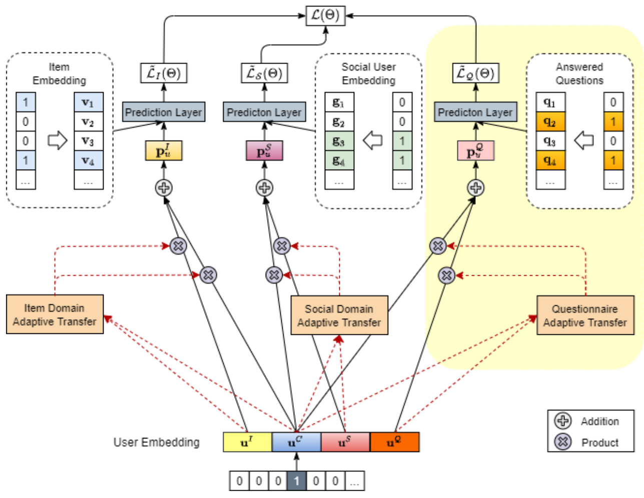


**Figure 1: High-level illustration of our proposed model. The area bounded by the yellow background marks the extension to EATNN.**

3. The entire model is jointly learned by optimizing the sum of losses. That is, the sum of each domain's loss.

## 4.2 Generating a Questionnaire

Since our proposed model uses answers to a questionnaire, we need to derive questions used in this questionnaire. In [19] Shi et al. propose a model called Local Representative-Based Matrix Factorization, LRMF, which has achieved great performance when recommending to extreme cold-start users. LRMF can be considered a state-of-the-art questionnaire-based approach and therefore, we decide to generate our questionnaires using this approach.

LRMF is an extension of Representative-Based Matrix Factorization, RBMF[15], which is a matrix factorization model but in which users are represented by their answers to selected representative questions. The extension of RBMF is essentially concerned with selecting better representative questions. More specifically, LRMF splits the representative questions into two: *global* and *local* representatives. Global representatives are used for dividing users into groups such that collective preferences are captured, while local representatives are used for capturing personalized preferences.

In short, LRMF tries to minimize:

$$\sum_{g \in \mathcal{G}} \{ ||\mathbf{R}^g - [\mathbf{U}_1^g; \mathbf{U}_2^g; \boldsymbol{e}]\mathbf{T}^g\mathbf{V}||_F^2 + \alpha||\mathbf{T}^g||_F^2 + \beta||\mathbf{V}||_F^2 \} \quad (1)$$

where $\mathcal{G}$ denotes the groups (subsets of users), $\mathbf{R}^g$ denotes group $g$'s interaction matrix, $\mathbf{U}_1^g$ and $\mathbf{U}_2^g$ denotes group $g$'s answers to global and local representatives respectively, $\boldsymbol{e}$ is a vector of ones, $\mathbf{T}^g$ is group $g$'s transformation matrix and $\mathbf{V}$ is the item representation matrix.

In order to minimize equation 1, Shi et al. propose an alternating least squares(ALS) like optimization strategy. That is, when learning one part, they fix the other parts. First, they learn global representatives by constructing a binary decision tree where nodes correspond to global representatives and contain a set of users. Each node has two children nodes: one containing users that has expressed a preference towards the asked question and one containing users who have not. The decision tree is recursively built until the depth of the tree matches the desired number of global representatives. Hereafter, they learn local representatives and transformation matrices for each group (leaf) in the decision tree by applying maxvol[9] and solving a Sylvester equation[2] respectively. Finally, they update the item representation, $\mathbf{V}$, by using a closed-form solution.

For a number of iterations or until the model has converged the above mentioned operations are applied to obtain a model. The obtained model contains, among other things, a questionnaire with global representatives and local representatives, which we will use later. Please note, that the questionnaire is dynamic, i.e. each user is potentially asked different questions, due to the nature of the decision tree.

## 4.3 Social-aware Recommendation

Following the assumption, that combining data from more domains better reflect users' preferences, we wanted to extend a social-aware approach with completed questionnaires obtained by LRMF. In [5] Chen et al. proposes a model called EATNN, which efficiently uses knowledge from the social domain in order to make better recommendations in the item domain. We decided to use EATNN as the basis of our model because of 1) the proven efficiency of the approach, 2) its ability to make use of both the item and social domain and 3) its flexibility in terms of used domains due to its joint optimization. Part of figure 1 depicts a high-level abstraction of EATNN.

As with our proposed model, EATNN transfers knowledge between domains by using attention modules, which excel at automatically learning relationships between domains. Formally, the attention of the item and social domains are defined as:

$$\begin{aligned}\alpha^*_{(C,u)} &= \mathbf{h}_\alpha^T\sigma(\mathbf{W}_\alpha\mathbf{u}^C + \mathbf{b}_\alpha); \quad \alpha^*_{(I,u)} = \mathbf{h}_\alpha^T\sigma(\mathbf{W}_\alpha\mathbf{u}^I + \mathbf{b}_\alpha)\\ \beta^*_{(C,u)} &= \mathbf{h}_\beta^T\sigma(\mathbf{W}_\beta\mathbf{u}^C + \mathbf{b}_\beta); \quad \beta^*_{(I,u)} = \mathbf{h}_\beta^T\sigma(\mathbf{W}_\beta\mathbf{u}^I + \mathbf{b}_\beta)\end{aligned} \tag{2}$$

where $\mathbf{W}_\alpha \in \mathbb{R}^{k\times d}$, $\mathbf{b}_\alpha \in \mathbb{R}^k$, $\mathbf{h}_\alpha \in \mathbb{R}^k$ are item domain specific parameters, $\mathbf{W}_\beta \in \mathbb{R}^{k\times d}$, $\mathbf{b}_\beta \in \mathbb{R}^k$, $\mathbf{h}_\beta \in \mathbb{R}^k$ are social domain specific parameters. $d$ is the dimension of embedding vectors, $k$ is the output dimension of attention modules and $\sigma$ is the $ReLU$ activation function[18], which comes with many advantages such as scale-invariance, efficient computation and gradients not vanishing. The attention scores are normalized with softmax to ensure values between 0 and 1:

$$\begin{aligned}\alpha_{(C,u)} &= \frac{\exp(\alpha^*_{(C,u)})}{\exp(\alpha^*_{(C,u)}) + \exp(\alpha^*_{(I,u)})} = 1 - \alpha_{(I,u)}\\ \beta_{(C,u)} &= \frac{\exp(\beta^*_{(C,u)})}{\exp(\beta^*_{(C,u)}) + \exp(\beta^*_{(S,u)})} = 1 - \beta_{(S,u)}\end{aligned} \tag{3}$$

Now, we use these attention weights on the user profiles in order to make use of the transferred knowledge. Formally, we define a new latent feature vector for a user in both the item and social domain:

$$\mathbf{p}_u^I = \alpha_{(I,u)}\mathbf{u}^I + \alpha_{(C,u)}\mathbf{u}^C; \quad \mathbf{p}_u^S = \beta_{(S,u)}\mathbf{u}^S + \beta_{(C,u)}\mathbf{u}^C \tag{4}$$

Finally, we need to make predictions of a user $u$'s preference towards an item $v$, $\hat{R}_{uv}$, and preference towards a friend $t$, $\hat{G}_{ut}$, such that we can compare predictions to ground truths and compute a loss. For this, an output layer is employed:

$$\hat{R}_{uv} = \mathbf{h}_I^T(\mathbf{p}_u^I \odot \mathbf{q}_v); \quad \hat{G}_{ut} = \mathbf{h}_S^T(\mathbf{p}_u^S \odot \mathbf{g}_t) \tag{5}$$

where $\mathbf{h}_I \in \mathbb{R}^d$ and $\mathbf{h}_S \in \mathbb{R}^d$ are the output layers for item and social domain respectively. $\mathbf{q}_v \in \mathbb{R}^d$ and $\mathbf{g}_t \in \mathbb{R}^d$ are latent vectors of item $v$ and friend $t$. $\odot$ denotes the element-wise product. Making recommendations to a user $u$ comes easy, we simply compute $\hat{R}_{uv}$ for all items and order the items descending based on their score.

When optimizing EATNN, Chen et al. suggest an efficient whole-data based learning strategy. However, we cannot directly employ this, since we work in an explicit setting and the strategy they suggest is aimed for implicit feedback. While we use the suggested loss function for the social domain, $\tilde{\mathcal{L}}_S(\Theta)$, shown in equation 7, we change the suggested loss function for the item domain to a simple squared error, because of its simplicity, in order to capture explicit feedback:

$$\tilde{\mathcal{L}}_I(\Theta) = \sum_{u\in\mathbf{B}}\sum_{v\in\mathbf{V}}(R_{uv} - \hat{R}_{uv})^2 \tag{6}$$

$$\begin{aligned}&\tilde{\mathcal{L}}_S(\Theta) =\\ &\sum_{i=1}^{d}\sum_{j=1}^{d}\left((h_{S,i}h_{S,j})\left(\sum_{u\in\mathbf{B}} p^S_{u,i}p^S_{u,j}\right)\left(\sum_{t\in\mathbf{U}} c_t^{S-} g_{t,i}g_{t,j}\right)\right)\\ &+\sum_{u\in\mathbf{B}}\sum_{t\in\mathbf{U}^+}\left((1-c_t^{S-})\hat{G}_{ut}^2 - 2\hat{G}_{ut}\right)\end{aligned} \tag{7}$$

where $c_t^{S-}$ are weights of negative samples suggested by [10]:

$$c_t^{S-} = c_0^S \frac{n_t^\rho}{\sum_{j=1}^{U} n_j^\rho}; n_t = \frac{|\mathcal{G}_t|}{\sum_{j=1}^{U}|\mathcal{G}_j|} \tag{8}$$

with $\mathcal{G}_t$ denoting the amount of social connections user $t$ has, $c_0^S$ denoting the overall weight of missing data and $\rho$ controls the significance of popular friends over unpopular ones.

Since we want to jointly optimize the model, we integrate both the item domain loss and social domain loss into one loss:

$$\mathcal{L}(\Theta) = \tilde{\mathcal{L}}_I(\Theta) + \mu\tilde{\mathcal{L}}_S(\Theta) \tag{9}$$

where $\mu$ is a parameter adjusting the influence of the social domain.

## 4.4 Final Model

With a questionnaire and the base model in place, we now, in more detail, look at our proposed model. First, we employ LRMF to obtain a questionnaire, and then we integrate a loss for the questionnaire, $\tilde{\mathcal{L}}_Q(\Theta)$, following the same logic as equation 7 into a total loss, such that our model effectively can use knowledge from this domain as well:

$$\mathcal{L}(\Theta) = \tilde{\mathcal{L}}_I(\Theta) + \mu(\tilde{\mathcal{L}}_S(\Theta) + \tilde{\mathcal{L}}_Q(\Theta)) \tag{10}$$

With this loss, we can learn our model in an end-to-end manner using an existing optimizer such as ADAM[12], AdaGrad[7] or alike.

$$\tilde{\mathcal{L}}_Q(\Theta) = \sum_{i=1}^{d}\sum_{j=1}^{d}\left((h_{Q,i}h_{Q,j})\left(\sum_{u\in\mathbf{B}} p^Q_{u,i}p^Q_{u,j}\right)\left(\sum_{q\in\mathbf{Q}} c^{Q-}_q q_{t,i}q_{t,j}\right)\right) + \sum_{u\in\mathbf{B}}\sum_{q\in\mathbf{Q}+}\left((1-c^{Q-}_q)\hat{Q}^2_{uq} - 2\hat{Q}_{uq}\right) \tag{11}$$

where $c^{Q-}_q$ is calculated similarly as shown in equation 8.

In order to transfer knowledge from the questionnaire into the item domain, we will use a similar attention module as for transferring knowledge between the item and social domain. Formally, we define it as:

$$\gamma^*_{(C,u)} = \mathbf{h}^T_\gamma \sigma(\mathbf{W}_\gamma \mathbf{u}^C + \mathbf{b}_\gamma); \gamma^*_{(Q,u)} = \mathbf{h}^T_\gamma \sigma(\mathbf{W}_\gamma \mathbf{u}^I + \mathbf{b}_\gamma) \tag{12}$$

where $\mathbf{W}_\gamma \in \mathbb{R}^{k\times d}$, $\mathbf{b}_\gamma \in \mathbb{R}^k$, $\mathbf{h}_\gamma \in \mathbb{R}^k$ are questionnaire specific parameters. We also apply softmax, as in equation 3, in order to obtain attention weights, $\gamma_{(C,u)}$ and $\gamma_{(Q,u)}$, between 0 and 1.

With $\gamma_{(C,u)}$ and $\gamma_{(Q,u)}$ we obtain user embeddings for the questionnaire domain similarly as for the item and social domain show in equation 4:

$$\mathbf{p}^Q_u = \gamma_{(Q,u)}\mathbf{u}^Q + \gamma_{(C,u)}\mathbf{u}^C; \tag{13}$$

The prediction layer for the questionnaire domain again follows the same logic as equation 5, and we get:

$$\hat{Q}_{uq} = \mathbf{h}^T_Q(\mathbf{p}^Q_u \odot \mathbf{q}_q) \tag{14}$$

where $\mathbf{h}_Q$ denotes an output layer in the questionnaire domain.

*Complexity.* In [5] it is shown that updating EATNN with one batch in the social domain runs in $O((|\mathbf{B}| + |\mathbf{U}|)d^2 + |\mathcal{G}_\mathbf{B}|d)$ where $\mathcal{G}_\mathbf{B}$ denotes the social connections of users in the batch $\mathbf{B}$. Similarly for the questionnaire domain, we can derive $O((|\mathbf{B}| + |\mathbf{Q}|)d^2 + |\mathcal{Q}_\mathbf{B}|d)$. Since we changed the loss of the item domain, we obtain a different time complexity than in [5] for this domain: $O((|\mathbf{B}||\mathbf{V}|)d)$. Thus, we end up with a total time complexity for one batch of: $O((2|\mathbf{B}| + |\mathbf{U}| + |\mathbf{Q}|)d^2 + (|\mathcal{G}_\mathbf{B}| + |\mathcal{Q}_\mathbf{B}| + |\mathbf{B}||\mathbf{V}|)d)$.

Naturally, this is a more costly complexity than in [5] because we extend the model. However, the complexity added by extending with a questionnaire domain is small in practice since number of questions in a batch, $|\mathcal{Q}_\mathbf{B}|$, is restricted to be small. Furthermore, the complexity is increased because we work in an explicit setting and therefore, cannot directly employ the proposed efficient whole-data based learning. This is a trade-off we have to take working in this setting.

## 4.5 Model Learning

To optimize our objective function, equation 10, we adopt a mini-batch ADAM[12] optimizer, because of its advantages: computational efficiency, not prone to exploding or vanishing gradients and is well suited for optimizing models with many parameters. In order to obtain batches of training samples, we first split users into batches. Then, for all batches, we conduct interviews with the questionnaire obtained with LRMF and use both all item interactions and social connections to form training samples.

Because our model is prone to overfitting, we employ dropout, which is effective at handling this. Specifically, we randomly drop $\phi$ percent of the transferred user embeddings, $\mathbf{p}^I_u$, $\mathbf{p}^S_u$ and $\mathbf{p}^Q_u$, with $\phi$ denoting the dropout probability.

# 5. EXPERIMENTS

In this section we will first describe the experimental setting, then we will perform an analysis on the obtained results. Furthermore, we make experiments in different settings and analyse these experiments.

## 5.1 Experimental Settings

### 5.1.1 Datasets

We experiment with two publicly available datasets: ***Ciao***[1] and ***EachMovie***[2]. Both of these datasets contains users' ratings to items they have rated in the form of $(u, i, r)$-triplets. ***Ciao*** comes with a social network, where ***EachMovie*** does not. Since we need a social network for our proposed model, we construct it such that each user, $u$, gets five (arbitrarily chosen) friends which are randomly drawn from a uniform distribution. While this most likely does not reflect reality since in social networks some users are *power users*, i.e. some users are more popular as friends, we limit ourselves to this social network for simplicity. Obviously, a more realistic social network is preferred and one approach could be to use similar distributions for selecting friends as the distributions in the social network of ***Ciao***. For both datasets, we preprocess them such that all items have at least five interactions. Table 2 shows statistical details of the two datasets after preprocessing. Where ***Ciao*** on average has 3.4 ratings per user and $\approx 17$ ratings per items, ***EachMovie*** are much more dense with $\approx 46$ ratings per user and $\approx 1,743$ ratings per item on average. The same pattern is apparent for the social connections, where ***EachMovie*** has 5 social connections per user where ***Ciao*** on average has $\approx 1$ social connection per user.

| | **Ciao** | **EachMovie** |
|---|---|---|
| **#Users** | 15,341 | 61,265 |
| **#Items** | 3,085 | 1,613 |
| **#Ratings** | 52,314 | 2,811,692 |
| **%Density** | 0.11% | 2.85% |
| **#Relationships** | 16,616 | 306,325 |

**Table 2: Statistical details on the used datasets.**

### 5.1.2 Baselines

We compare our proposed model to the following baselines:

- **LRMF**[19]: This MF approach represents users through answers to a derived questionnaire (explained in section 4.2) and items in a latent space. Predictions are

[1]https://www.librec.net/datasets.html
[2]http://www.gatsby.ucl.ac.uk/ chuwei/ data/EachMovie/eachmovie.html

obtained by performing a dot-product of user representations and item latent factors.

- **EATNN**[5]: The basis of our model described in detail in section 4.3.
- **BPR**[20]: An MF model, which optimizes the *Bayesian Personalized Ranking* objective function.
- **SBPR**[22]: This method is an extension of BPR, which assumes users like items they have consumed more than items their friends have consumed, which they like more than items they have not consumed.
- **QSBPR**: An extension of SBPR, which we have tailored ourselves, based on the assumption that users like items they have consumed more than items they say they like (in the questionnaire) than items their friends have consumed than items the users have not consumed.

Since we extend EATNN with a questionnaire obtained by LRMF, we compare our model to both EATNN and LRMF. The other baselines are chosen because Chen et al. compares their approach, EATNN, to them. Naturally, because we extend EATNN we, optimally, also outperform their baselines. Because of time restrictions, we have not compared with all of EATNN's baselines and their respective questionnaire extended versions.

### 5.1.3 Evaluation Metrics

We adopt three metrics to evaluate the performance of our model: $NDCG@K$, $Precision@K$ and $Recall@K$.

$$\text{NDCG@K} = \frac{\text{DCG@K}}{\text{IDCG@K}} \tag{15}$$

where

$$\text{DCG@K} = \frac{1}{|\mathbf{U}|} \sum_{u \in \mathbf{U}} \sum_{i=1}^{K} \frac{2^{\text{rel}_i} - 1}{\log_2(i+1)}$$

and IDCG@K is equal to the above equation but computed with the optimal top-K list. $\text{rel}_i$ denotes the relevance of item $i$ which is 1 if the user rated the item or 0 if not to make fair comparisons with baselines that are intended for implicit feedback where an item either is relevant or not. If we were to only compare with baselines intended for explicit feedback, one can change $\text{rel}_i$ such that it equals the actual rating of item $i$, i.e. $R_{ui}$.

$$\text{Precision@K} = \frac{\sum_{u \in \mathbf{U}} \#\text{tp}}{\sum_{u \in \mathbf{U}} K} \tag{16}$$

$$\text{Recall@K} = \frac{\sum_{u \in \mathbf{U}} \#\text{tp}}{\sum_{u \in \mathbf{U}} \#\text{tp} + \#\text{fn}} \tag{17}$$

where $\mathbf{U}$ denotes the set of users, #tp denotes the number of items in the top-K recommended list that user $u$ has rated and #fn denotes the number of items in the top-K recommended list that user $u$ has not rated.

### 5.1.4 Experimental Details

In order to simulate the cold-start scenario, we split our dataset into 25% training and 75% test, such that 25% of each user's ratings are used for training and the remaining 75% are used for test. This will simulate several degrees of the cold-start problem, even the extreme case, where a user has zero training interactions.

The parameters of all baselines are initialized according to the corresponding paper. For our model we manually tuned the batch size, $|\mathbf{B}|$, in [64, 128, 256], the output dimension of attention modules, $k$, in [16, 32, 64], number of latent factors, $d$, in [32, 64, 128], learning rate, $lr$, in [0.001, 0.002, 0.005] and dropout probability, $\phi$, in [0.1, 0.3, 0.7]. After tuning, we found the best parameters as follow: $|\mathbf{B}| = 128$, $k = 32$, $d = 64$, $lr = 0.001$ and $\phi = 0.1$. Finally, we set $\mu = 0.1$ meaning the social and questionnaire domain both influence 10% of the total loss.

Since BPR, SBPR and QSBPR works with implicit feedback, we transform the data such that $(u, i, r)$-triplets become $(u, i)$-tuples, where $u$ is a user, $i$ is an item and $r$ is the given rating.

## 5.2 Comparative Analysis Of Performance

The results of QEATNN and the described baselines on the two datasets are shown in table 3. To evaluate performance for different tasks, we experiment with different length of the recommended list ($K = 10, 50, 100$). From the results, the following observations can be made:

First, for ***Ciao*** we see models utilizing both questionnaires and social connections (QSBPR and QEATNN) generally perform worse than their respective models utilizing only social connections (SBPR and EATNN). Please recall, that we obtain our questionnaire by running LRMF. Looking at the performance of LRMF, we see a relatively poor performance compared to the other models, which can explain the poor performance of the questionnaire-based models. This can be the case, since poorly chosen questions may move users' preferences away from their actual preferences. QSBPR performs significantly worse than SBPR, which can be explained by 1) the poorly generated questionnaire as mentioned and 2) the assumption, that users like items they have interacted with, more than items they say they like, more than items their friends have interacted with, more than items they have not interacted with, does not hold. In fact, we might not ask about items that users like, and with this the assumption should maybe be moved around. However, QSBPR cannot handle this. Finally, BPR performs best on NDCG, while SBPR performs best on precision and recall. Diving into the number of training samples for each user, we see that in average a user has $\approx 0.65$ ratings and $\approx 1.08$ social connections. That means, some of our users must be extreme cold in terms of item interactions and other users are very cold. Back to the results, this suggest that BPR and SBPR can learn better with less data than the other models.

Second, we see an improve in performance for all models on ***EachMovie*** compared to their performance on ***Ciao***. This can be explained by the difference in size of the two datasets. ***EachMovie*** is more dense both in terms of item

| *Ciao* | NDCG@10 | NDCG@50 | NDCG@100 | Prec@10 | Prec@50 | Prec@100 | Recall@10 | Recall@50 | Recall@100 |
|---|---|---|---|---|---|---|---|---|---|
| **BPR** | **0.0228** | **0.0430** | **0.0542** | **0.0081** | 0.0056 | 0.0043 | **0.0394** | 0.1271 | 0.1843 |
| **SBPR** | 0.0154 | 0.0389 | 0.0514 | **0.0081** | **0.0058** | **0.0045** | 0.0365 | **0.1288** | **0.1914** |
| **QSBPR** | 0.0022 | 0.0064 | 0.0107 | 0.0012 | 0.0012 | 0.0012 | 0.0036 | 0.0195 | 0.0410 |
| **LRMF** | 0.0049 | 0.0049 | 0.0049 | 0.0018 | 0.0011 | 0.0009 | 0.0107 | 0.0296 | 0.0510 |
| **EATNN** | 0.0155 | 0.0280 | 0.0354 | 0.0069 | 0.0045 | 0.0035 | 0.0238 | 0.0719 | 0.1078 |
| **QEATNN** | 0.0160 | 0.0273 | 0.0353 | 0.0056 | 0.0035 | 0.0028 | 0.0234 | 0.0654 | 0.1022 |
| *EachMovie* | **NDCG@10** | **NDCG@50** | **NDCG@100** | **Prec@10** | **Prec@50** | **Prec@100** | **Recall@10** | **Recall@50** | **Recall@100** |
| **BPR** | 0.4888 | 0.5705 | 0.6169 | 0.4412 | 0.3026 | 0.2147 | 0.2577 | **0.6601** | **0.8028** |
| **SBPR** | 0.4597 | 0.5112 | 0.5644 | 0.4058 | 0.2761 | 0.1834 | 0.2241 | 0.6289 | 0.7549 |
| **QSBPR** | 0.0484 | 0.0499 | 0.0521 | 0.0387 | 0.0256 | 0.0170 | 0.0206 | 0.0607 | 0.0737 |
| **LRMF** | 0.2811 | 0.2662 | 0.2595 | 0.2891 | 0.2319 | 0.1776 | 0.1845 | 0.5507 | 0.7199 |
| **EATNN** | **0.5391** | **0.6067** | **0.6488** | **0.4763** | **0.3176** | **0.2204** | **0.2731** | 0.6572 | 0.7905 |
| **QEATNN** | 0.5132 | 0.5789 | 0.6204 | 0.4618 | 0.3108 | 0.2165 | 0.2473 | 0.6228 | 0.7545 |

**Table 3: Performance of our proposed model and baselines on *Ciao* and *EachMovie*. For each metric, the best one is marked with bold.**

interactions and social connections (please recall, the way we generate our social network). Obviously, with more data available, we should be able to recommend more precisely, which is exactly what we see.

Third, for ***EachMovie*** the pattern is a little different than for ***Ciao***. Here, EATNN outperforms every other baselines except on *Recall@50* and *Recall@100* where SBPR performs better. This suggests, that EATNN efficiently manages to learn users' preferences by adaptively transferring knowledge from both the item domain and the social domain (even though it is a quiet random social network). The fact that EATNN performs as relatively well as it does, suggests that rather than using actual friends it can work well by simply having other users to draw knowledge from. This is not the case for SBPR, which is outperformed by BPR, which can be explained by SBPR's assumption not holding. Since the social network is naively constructed, it makes good sense, that the assumption does not hold. More specifically, with this naively constructed social network, we cannot be sure that the items a user's friends have interacted with are more preferred of that user than items he/she has not interacted with. Furthermore, while QSBPR and QEATNN do not outperform their base models (SBPR and EATNN) we see a more comparable performance than for ***Ciao***. This can be explained by the better performance of LRMF, which in turn provides better questionnaires for these two models.

Finally, even though our model, QEATNN, does not outperform its base-model EATNN, we see comparable performance on both datasets. In fact, on ***Ciao*** we are better on *NDCG@10*. It might be the case, that QEATNN is better when the task is to recommend shorter lists than EATNN. Therefore, we conduct a small side experiment, where we compute NDCG on shorter lists ($K = 1, 2, 5$) for EATNN and QEATNN. Figure 2 shows these results. We see that for the ***Ciao*** dataset, we outperform EATNN on shorter lists, where for the ***EachMovie*** dataset, EATNN outperforms our model. This suggest, that on one hand with less data available in the item and social domain (as in ***Ciao***), we gain performance by adding knowledge through a questionnaire, even though it is a relatively poor questionnaire. On the other hand, when we have more data available in the item and social domain (as in ***EachMovie***), we add noise, which can be explained by the questionnaire not being very good.

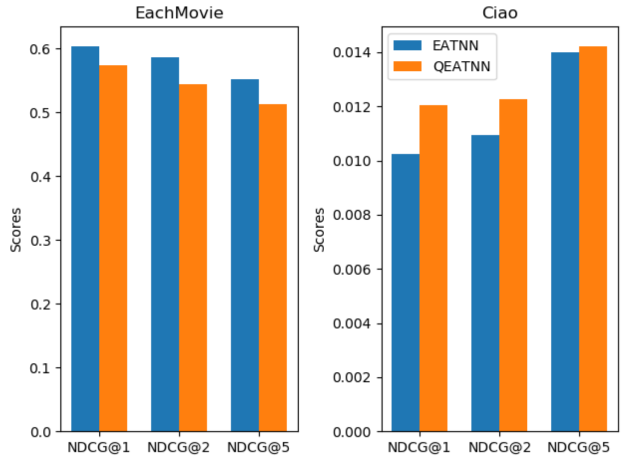


**Figure 2: Performance of EATNN and QEATNN on shorter lists ($K = 1, 2, 5$).**

## 5.3 Are Users Actually Cold?

In this section, we conduct experiments where we more thoroughly investigate the existence of cold users. In section 5.1.4 we described the protocol for simulating the cold-start scenario. That is, we randomly split each user's interactions into 25% train and 75% test. For the experiments conducted in this section, we consider users with 10 or less ratings as cold. Analysing the number of interactions per user in the training set, we found that for both ***Ciao*** and ***EachMovie*** we have users who are not cold, and for ***EachMovie*** we have many users who are not cold. Figure 3 shows histograms of number of ratings vs number of users. This allows us to evaluate the performances of the models on cold user in an otherwise warm environment (using ***EachMovie***) and in an almost completely cold environment (using ***Ciao***).

While figure 3 can explain the differences in performance on the two datasets presented in table 3, because users in general have more interactions in the ***EachMovie*** dataset, the results on ***EachMovie*** might not in fact reflect an answer to our problem definition from section 1 due to the sheer amount of interaction data. Therefore, we tested our model's performance as well as baselines' performance when the task is to alleviate the cold-start problem. More specif-

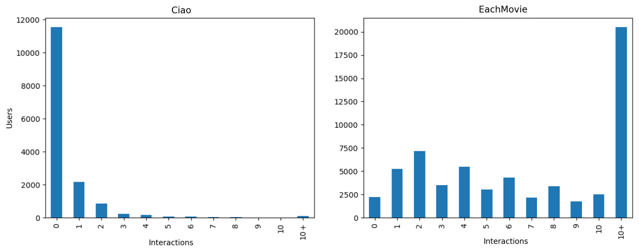


**Figure 3: Histogram over number of ratings and users. The $x$-axis denotes number of ratings and the $y$-axis denotes number of users.**

ically, we split users from the training sets into groups, such that groups contain users who have the same amount of ratings in the training data, and test performance with *NDCG@10* on each of these groups. That means, we get 11 groups: $g_0, g_1, g_2, ..., g_{10}$ where $g_x$ denotes the group of users who have $x$ ratings in the training set. Results are displayed in figure 4.

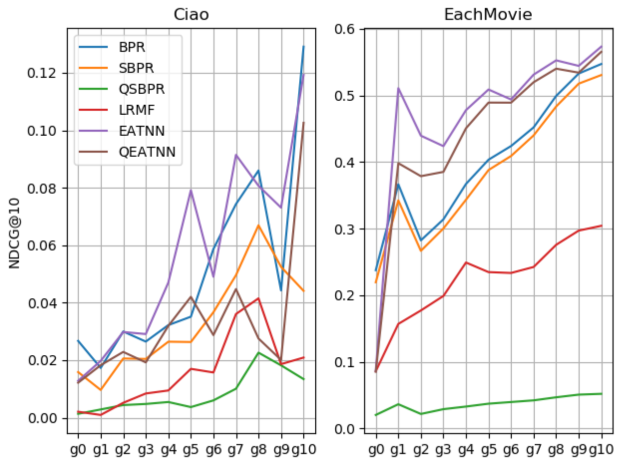


**Figure 4: *NDCG@10* for groups of users on the *Ciao* and *EachMovie* datasets. $g_x$ denotes the group of users with $x$ ratings.**

Figure 4 shows that both datasets generally have a lower performance at the groups with a lower amount of interactions. This, intuitively, makes sense as less data means less knowledge about user preferences, which results in worse recommendations. However, on the ***EachMovie*** dataset we see most models perform better at lower groups compared on the ***Ciao*** dataset. This is expected with the difference in size and density between the datasets. We also see that all the models perform considerably better on the ***EachMovie*** dataset compared to the ***Ciao*** dataset, which indicates that the models thrive better on larger datasets. This is especially true for the QEATNN model, that almost rivals EATNN on the ***Each Movie*** dataset.

## 5.4 Does the Social Network Influence Performance?

Working with models utilizing social data, we, in this section, wanted to measure our models' ability to make use of this social data and if a fake, i.e. generated, social network could be used instead of a real one. If this is the case, our model can be used in cases where social data is not available. Furthermore, it can easily be the case, that social networks have users who do not have any friends, and it may be that extending the real social network can improve performance.

In order to conduct experiments in these settings we generate 4 social networks for the ***Ciao*** dataset, because we have a real social network which we can compare with. We generate the social networks as follows:

1. *random5*: every user gets 5 random friends selected by a uniform distribution.
2. *random10*: every user gets 10 random friends selected by a uniform distribution.
3. *power23*: users who do not have any friends gets 23 friends selected by a *power* distribution. We define the power distribution, such that the probability for selecting user $u$ equals the normalized frequency of $u$ as a friend in the real social network. This way, some users should become *power* users, i.e. users with high influence. We select 23 users, because in the real social network users have, in average, 23 friends.
4. *power5*: every user gets 5 friends selected by the power distribution described above.

We run EATNN and QEATNN with these generated social networks and compare performance, in terms of *NDCG@10*, with EATNN and QEATNN run with the real social network. We follow the same experimental settings as described in section 5.1.4. Figure 5 shows *NDCG@10* when using the 4 generated social networks, as well as the real one.

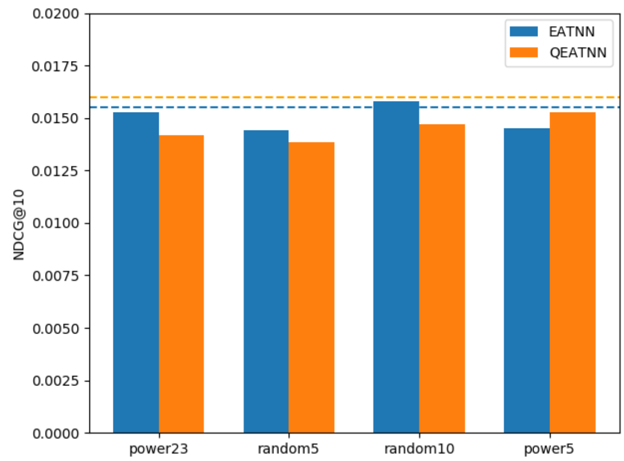


**Figure 5: *NDCG@10* for EATNN and QEATNN with different generated "fake" social networks. The dashed lines denotes the performance of EATNN and QEATNN with the provided social networks in the dataset.**

Looking at figure 5 we have the following observations. First, none of the generated networks improve performance for either QEATNN or EATNN, except for *random10* for EATNN.

This might suggest, that a more dense social network improve performance of EATNN. However, using *power23*, which can also be considered a dense social network, for EATNN does not improve performance, and therefore, it is more likely that *random10* improves performance because of its randomness.

Second, QEATNN is not outperformed when using any of the generated social networks. This suggests that using a more dense network is not optimal for QEATNN. This can be explained by, using a generated social network might influence QEATNN in a way such that the questionnaire domain takes over and thereby the knowledge in the social domain is neglected even though there might be some knowledge in it.

Finally, EATNN outperforms QEATNN when using all of the generated social networks except for *power5*. This can be the case, when QEATNN efficiently balances the weight of the social domain and the questionnaire domain. Each user having 5, to some degree, sophisticated chosen friends might be optimal when generating a social network for QEATNN in terms of its ability to balance the influence of the social domain and questionnaire domain. On the contrary, using the other generated social networks suggest that QEATNN struggles to balance the weights of the questionnaire and social domain. A further analysis of this could be done by looking at the weights coming out of QEATNN using the different generated social networks.

## 5.5 Performance on Extreme-Cold Start Users

As we have talked about previously, cold users provide few item interactions during training, which can be used for learning user preferences. An even more challenging task is, when users are extreme cold meaning they provide no item interactions which can be used for learning user preferences. In this section we will experiment with our proposed model's and the baselines' ability to recommend to extreme cold users. More specifically, we split our datasets into 70% train and 30% test, such that a user is either in train or test. That way, we simulate the extreme cold start case for users. The performance of the different baselines along with our proposed model are presented in table 4.

From the results in table 4 we can observe the following: First, for the ***Ciao*** dataset, we can see that SBPR, which uses data from the social domain outperform its base model, BPR. This makes sense, if we consider the extreme cold-start scenario, where users provide no item interactions. Thus, BPR, which rely solely on ratings in order to learn user preferences is expected to struggle. This is in line with previous work[4, 21, 22] which has suggested that social information reflects preferences of users. Furthermore, SBPR outperforms every other baseline except on recall where LRMF is the better. Even though, BPR is outperformed by SBPR, we see that BPR outperform our proposed model and the other baselines, except on recall where LRMF is the better, which makes use of the social domain and/or questionnaire domain. While BPR is a simple model, the results on the ***Ciao*** dataset suggest that BPR can better recommend towards extreme cold-start users than QSBPR, LRMF, EATNN and QEATNN. This can be explained by these methods being rather complex and since the size of the ***Ciao*** dataset is relatively small, they might not have enough knowledge from which they can learn user preferences.

Second, for the ***EachMovie*** we see that LRMF perform best on every metric. LRMF is intended for handling extreme cold-start users and this becomes evident here. Since LRMF performs as well as it does, one could hope, that the performance of our model, which makes use of the questionnaire in LRMF, keeps up with LRMF. However, this is not the case and is explained by our model not being intended for the extreme cold-start setting. While our model does not perform as well as LRMF, we outperform EATNN on every metric and the other baselines on *NDCG@10*, which most likely is because we use a well-suited questionnaire for learning user preferences. The fact that our model does not outperform LRMF is explained by the social network, which we generated ourselves. This is also the reason as to why BPR performs better than SBPR and QSBPR.

Finally, we see that all models perform much better on the ***EachMovie*** dataset than on the ***Ciao*** dataset. This is expected when considering the size and density of the two datasets. Intuitively, with more data we can better learn user preferences.

## 5.6 Working with Implicit Feedback

Since the base model of QEATNN is intended for implicit feedback, we wanted to measure our model's and the baselines' performance in an implicit feedback setting. For this, we experiment with the same implicit dataset as they do in EATNN[3]. This dataset also comes from Ciao. Statistical details on the dataset is shown in table 5.

The experiments in this section follow the same experimental protocol as described in section 5.1.4. That means, for each user we split his/her interactions into 25% train and 75% test and use all data available in the social domain. Figure 6 shows our proposed model's and the baselines' performance.

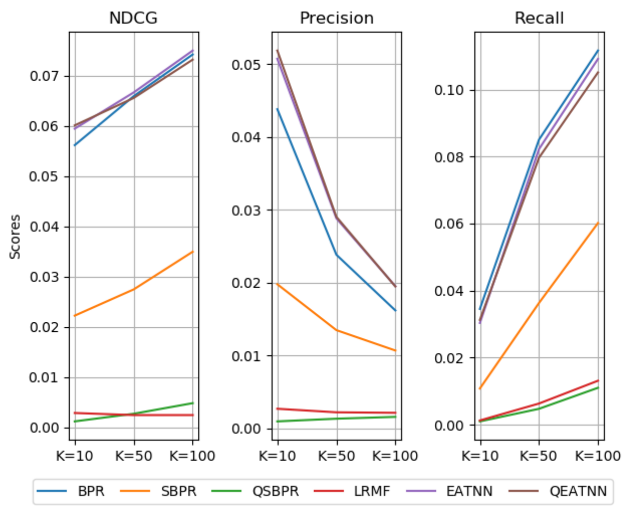


**Figure 6: Performance on the implicit dataset presented in table 5.**

[3]https://github.com/chenchongthu/EATNN/tree/master/data/ciao

| *Ciao* | NDCG@10 | NDCG@50 | NDCG@100 | Prec@10 | Prec@50 | Prec@100 | Recall@10 | Recall@50 | Recall@100 |
|---|---|---|---|---|---|---|---|---|---|
| **BPR** | 0.0249 | 0.0485 | 0.0622 | 0.0088 | 0.0065 | 0.0052 | 0.0398 | 0.1304 | 0.1983 |
| **SBPR** | **0.0277** | **0.0501** | **0.0642** | **0.0112** | **0.0070** | **0.0056** | 0.0535 | 0.1383 | 0.2080 |
| **QSBPR** | 0.0033 | 0.0084 | 0.0140 | 0.0018 | 0.0015 | 0.0016 | 0.0069 | 0.0262 | 0.0543 |
| **LRMF** | 0.0148 | 0.0148 | 0.0148 | 0.0084 | 0.0048 | 0.0036 | **0.0633** | **0.1658** | **0.2463** |
| **EATNN** | 0.0196 | 0.0336 | 0.0415 | 0.0070 | 0.0043 | 0.0033 | 0.0311 | 0.0862 | 0.1264 |
| **QEATNN** | 0.0216 | 0.0327 | 0.0394 | 0.0076 | 0.0048 | 0.0031 | 0.0320 | 0.0768 | 0.1100 |
| *EachMovie* | **NDCG@10** | **NDCG@50** | **NDCG@100** | **Prec@10** | **Prec@50** | **Prec@100** | **Recall@10** | **Recall@50** | **Recall@100** |
| **BPR** | 0.2197 | 0.3598 | 0.4408 | 0.3612 | 0.2158 | **0.1746** | 0.1863 | 0.4111 | 0.5640 |
| **SBPR** | 0.2104 | 0.3475 | 0.4269 | 0.3569 | 0.2098 | 0.1699 | 0.1815 | 0.4087 | 0.5485 |
| **QSBPR** | 0.0204 | 0.0332 | 0.0413 | 0.0341 | 0.0194 | 0.0157 | 0.0169 | 0.0396 | 0.0527 |
| **LRMF** | **0.3875** | **0.4168** | **0.4259** | **0.3751** | **0.2253** | 0.1539 | **0.2869** | **0.6604** | **0.8008** |
| **EATNN** | 0.2476 | 0.2773 | 0.3125 | 0.2268 | 0.1706 | 0.1321 | 0.1026 | 0.2889 | 0.3969 |
| **QEATNN** | 0.2550 | 0.2858 | 0.3242 | 0.2347 | 0.1775 | 0.1387 | 0.1035 | 0.3013 | 0.4177 |

**Table 4: Performance of our proposed model and baselines in the extreme cold-start setting. For each metric, the best one is marked with bold.**

| #Users | #Items | #Ratings | %Density | #Relationships |
|---|---|---|---|---|
| 7,267 | 11,211 | 157,995 | 0.19% | 111,781 |

**Table 5: Statistical details of the dataset with implicit feedback.**

From figure 6 we see that BPR, EATNN and QEATNN outperforms the other models on every metric. Common for these three models is that they perform very similar on every metric. This implies, that a combination of item interactions, social data and answered questionnaires does not reflect user preferences better than only item interactions. This can be the case, or it can be the case that EATNN and QEATNN are not able to learn from the social and questionnaire domain. The reason, that we can make these observations, is due to the fact that BPR neither utilizes the social or questionnaire domain.

Furthermore, we see SBPR performing worse than beforementioned, which again can indicate the social domain not reflecting user preferences. Finally, we see that LRMF and QSBPR are performing relatively poor compared to the other models. This is explained by LRMF not being intended for 1) cold-start setting and 2) for implicit feedback. To this end, we can expect QSBPR to perform poor as well.

# 6. CONCLUSION AND FUTURE WORK

In this work, we propose *Questionnaire-Based Efficient Adaptive Transfer Neural Network*, QEATNN, to alleviate the cold-start user problem. QEATNN is an extension of EATNN, in which, as well as item and social domains, we also utilize questionnaires. By utilizing knowledge in the three domains, QEATNN has more knowledge about cold-users preferences. We obtain a dynamic questionnaire by applying a state-of-the-art questionnaire-based approach, *Local Representative-Based MF*, LRMF.

We have performed extensive experiments on two publicly available datasets. Furthermore, we have performed experiments on another dataset to validate QEATNN's ability to work with implicit feedback. While QEATNN does not outperform baselines in every setting, results show that QEATNN perform comparably with state-of-the-art approaches.

Since our model needs a questionnaire, which in itself is a task, this could be integrated in the learning of the model instead of relying on a questionnaire produced by another model. This might affect our model's time needed for learning but in turn might generate a more model-specific questionnaire and thus, give better recommendations. Doing this, QEATNN becomes applicable in more settings, as it becomes invariant to the limitations of the model producing the questionnaire.

Furthermore, because of our model's linear objective function, we cannot learn non-linear dependencies of the domains. Therefore, in the future, we are interested in extending our optimization, such that it can handle non-linearity. This is also proposed as future work by the authors of EATNN[5].

Finally, Since LRMF performs as well as it does for extreme cold-start users and because we see that social data reflect users preferences, we are interested in extending LRMF with social data. This might improve the performance of LRMF in this setting.


# ACKNOWLEDGEMENTS

We thank our supervisor, Peter Dolog, for his valuable comments and suggestions provided during this work (Februar 2020-June 2020). Also, we thank Anders Brams, Anders Langballe and Theis Jendal, another master thesis group working with Recommender Systems as well, for their professional back-and-forth.


# References


[1] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1409.0473`.

[2] R. H. Bartels and G. W. Stewart. Solution of the matrix equation ax + xb = c [f4]. *Commun. ACM*, 15 (9):820–826, Sept. 1972. ISSN 0001-0782. doi: 10.1145/361573.361582. URL `https://doi.org/10.1145/361573.361582`.

[3] C. Chen, M. Zhang, Y. Liu, and S. Ma. Neural attentional rating regression with review-level explanations. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, page 1583–1592, Republic and

Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee. ISBN 9781450356398. doi: 10.1145/3178876.3186070. URL https://doi.org/10.1145/3178876.3186070.

[4] C. Chen, M. Zhang, Y. Liu, and S. Ma. Social attentional memory network: Modeling aspect-and friend-level differences in recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 177–185. ACM, 2019.

[5] C. Chen, M. Zhang, C. Wang, W. Ma, M. Li, Y. Liu, and S. Ma. An efficient adaptive transfer neural network for social-aware recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 225–234, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450361729. doi: 10.1145/3331184.3331192. URL https://doi.org/10.1145/3331184.3331192.

[6] C. B. Do and A. Y. Ng. Transfer learning for text classification. In *Proceedings of the 18th International Conference on Neural Information Processing Systems*, NIPS'05, page 299–306, Cambridge, MA, USA, 2005. MIT Press.

[7] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12(null):2121–2159, July 2011. ISSN 1532-4435.

[8] C. Gao, X. Chen, F. Feng, K. Zhao, X. He, Y. Li, and D. Jin. Cross-domain recommendation without sharing user-relevant data. In *The World Wide Web Conference*, WWW '19, page 491–502, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450366748. doi: 10.1145/3308558.3313538. URL https://doi-org.zorac.aub.aau.dk/10.1145/3308558.3313538.

[9] S. Goreinov, I. Oseledets, D. Savostyanov, E. Tyrtyshnikov, and N. Zamarashkin. How to find a good submatrix. *Matrix Methods: Theory, Algorithms and Applications*, 04 2010. doi: 10.1142/9789812836021_0015.

[10] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, page 549–558, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450340694. doi: 10.1145/2911451.2911489. URL https://doi.org/10.1145/2911451.2911489.

[11] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*, pages 263–272, 2008.

[12] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2014.

[13] Y. Koren. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, page 426–434, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605581934. doi: 10.1145/1401890.1401944. URL https://doi.org/10.1145/1401890.1401944.

[14] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8): 30–37, 2009.

[15] N. N. Liu, X. Meng, C. Liu, and Q. Yang. Wisdom of the better few: Cold start recommendation via representative based rating elicitation. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, page 37–44, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450306836. doi: 10.1145/2043932.2043943. URL https://doi.org/10.1145/2043932.2043943.

[16] H. Ma. On measuring social friend interest similarities in recommender systems. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '14, page 465–474, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450322577. doi: 10.1145/2600428.2609635. URL https://doi.org/10.1145/2600428.2609635.

[17] D. S. Maitra, U. Bhattacharya, and S. K. Parui. Cnn based common approach to handwritten character recognition of multiple scripts. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1021–1025, 2015.

[18] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[19] L. Shi, W. X. Zhao, and Y.-D. Shen. Local representative-based matrix factorization for cold-start recommendation. *ACM Trans. Inf. Syst.*, 36(2), Aug. 2017. ISSN 1046-8188. doi: 10.1145/3108148. URL https://doi.org/10.1145/3108148.

[20] Z. G. Steffen Rendle, Christoph Freudenthaler and L. Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. *CoRR*, abs/1205.2618, 2012. URL http://arxiv.org/abs/1205.2618.

[21] L. Xiao, Z. Min, Z. Yongfeng, L. Yiqun, and M. Shaoping. Learning and transferring social and item visibilities for personalized recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 337–346, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450349185. doi: 10.1145/3132847.3132911. URL https://doi.org/10.1145/3132847.3132911.

[22] T. Zhao, J. McAuley, and I. King. Leveraging social connections to improve personalized ranking for collaborative filtering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, page 261–270, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450325981. doi: 10.1145/2661829.2661998. URL https://doi.org/10.1145/2661829.2661998.

[23] K. Zhou, S.-H. Yang, and H. Zha. Functional matrix factorizations for cold-start recommendation. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, page 315–324, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450307574. doi: 10.1145/2009916.2009961. URL `https://doi.org/10.1145/2009916.2009961`.

# SUMMARY

In this paper we attempt to tackle the cold-start user problem, a notorious problem within the field of recommender systems. A lack of auxiliary user information forces us to be creative when trying to overcome this problem. Popular approaches include questionnaire-based approaches that utilizes an initial interview phase to gain more information about the user. Recently, methods utilizing social information has also seen an increase in popularity, as the social data becomes more and more available in this modern time and the empirical evidence that social data reflects users preferences.

We try to combine these two approaches, social-aware and questionnaire-based approaches, and utilize the advantages of both domains, a sort of *best of both worlds* solution. We do this by extending an already existing model called *Efficient Adaptive Transfer Neural Network*, with a questionnaire, and utilize its ability to balance the influence of the domains, depending on the available data. We name our model *Questionnaire-Based Efficient Adaptive Transfer Neural Network*, QEATNN. Intuitively, this would allow the model to utilize more information and better learn users' preferences which would result in better recommendations. QEATNN utilizes attention modules to learn how the three domains (item, social and questionnaire domain) should be balanced such that we obtain better recommendation. The model is learned by optimizing a joint loss of all three domains.

To obtain a questionnaire, we employ a state-of-the-art questionnaire-based approach, *Local Representative-Based Matrix Factorization*(LRMF). As with any other MF models, LRMF makes personalized recommendations by performing a dot-product of a user-specific vector with the item-specific vectors. The user-specific vectors are obtained by taking the users' interactions on the questions in the questionnaire. The questionnaire is obtained by running a proposed algorithm which results in a decision tree where nodes correspond to questions and each node has two children: *like* and *dislike*. That way, a dynamic questionnaire is derived, i.e. each user, potentially, has to answer different questions.

We experiment with our model and state-of-the-art baselines in a series of different circumstances, these being: the cold-start scenario, the extreme cold-start scenario, cold-start on implicit data, cold-start in a warm scenario and different types of generated simulations of social networks. We measure performance by computing widely used metrics: *NDCG*, *Precision* and *Recall*. Among the interesting results we see that randomly generated social network showed improvement on the EATNN model, compared to the actual social network accompanying the dataset.

Surprisingly we see our extension only managing to outperform its original model as well as our other baselines in some cases, specifically on short lists in the cold-start scenario, i.e. when users provide little knowledge about preferences of items, as opposed to the extreme cold-start scenario, where other approaches are superior.

Finally, we suggest to integrate the learning of a questionnaire, such that the questionnaire becomes more specific for our model, in the future. While this may increase computational complexity of our model, it could improve performance and make our model applicable in more settings as it becomes invariant to the limitations of the model producing the questionnaire.