MŰEGYETEM 1782

Context-aware factorization methods for implicit feedback based recommendation problems

Thesis booklet

Balázs Hidasi

Under the supervision of
Dr. Domonkos Tikk, Dr. Gábor Magyar

Budapest University of Technology and Economics, Hungary
Department of Telecommunications and Mediainformatics

May, 2015.

Contents

1 Introduction

Recommender systems are information filtering tools that help users in information overload to find interesting items (products, content, etc). Users get personalized recommendations that contain typically a few items deemed to be of user's interest. The relevance of an item with respect to a user is predicted by recommender algorithms; items with the highest prediction scores are displayed to the user.

The core of the recommender system is the recommender algorithm that ranks the items for the users based on their relevance. Recommender algorithms are usually sorted into five main approaches [13], out of which the two most prominent strategies are collaborative filtering (CF) and content-based filtering (CBF). CF algorithms use only the user–item interactions (also called events or transactions). The assumption of CF is that two users are similar if they consumed similar items; and two items are similar if they have been consumed by similar users [15]. CBF algorithms use item metadata (e.g. author, genre, etc.). First, the metadata of the items is analyzed using text mining and methods from information retrieval [2]. User profiles are built from the metadata of items the user liked/disliked using machine learning. Preferences are predicted by matching the user profile with item metadata [9]. CF algorithms are the most accurate amongst the pure approaches in a generic situation, e.g. they are more accurate than CBF methods if sufficient preference data is available [10].

CF algorithms can be classified into memory-based and model-based ones. The former are neighbor methods that make use of item or user rating vectors to define similarity, and they calculate recommendations as a weighted average of similar item or user rating vectors (e.g. [5, 12, 7]). In the last decade, model-based methods gained enhanced popularity, because they were found to be much more accurate in the Netflix Prize [4], a community contest launched in late 2006 that provided the largest explicit benchmark data set (100M ratings) for a long time. Model-based methods build generalized models that intend to capture user preference. The most successful approaches are the latent factor algorithms. These represent each user and item as a feature vector in a K dimensional latent feature space.

Matrix factorization: The most well-known latent feature based algorithms are matrix factorization (MF) methods (e.g. [3, 16, 11, 17, 6, 14]). Matrix factorization methods organize ratings or preferences into a matrix (R), whose dimensions are the users and the items. If user u rates item i with a rating r then $R_{u,i} = r$. R is a large matrix, but very sparse. The concept behind matrix factorization is to approximate R as the product of two low rank matrices ($\hat{R} = (M^{(U)})^T M^{(I)}$), referred to as feature matrices. One of the feature matrices belongs to the users ($M^{(U)}$) and the other to the items ($M^{(I)}$). The u^{th} row of the user feature matrix is the latent feature vector (of K length) for user u. Latent feature vectors are also assigned to items in a similar fashion. The predicted rating/preference of user u on item i is the dot product of their feature vectors, i.e. $\hat{R}_{u,i} = (M_u^{(U)})^T M_i^{(I)}$.

Feedback types: Depending on the nature of the user–item interactions, recommendation problems can be classified into explicit and implicit feedback based problems. Explicit feedback is provided by the users, usually in the form of ratings, and it explicitly encodes their preferences on the items. The classic explicit feedback based task is rating prediction, where the goal of the algorithm is to accurately estimate missing ratings of the users on items. The goal of a recommender system however is to present a small number of items to the users that are interesting/useful to them. For this the recommender has to rank the items first (based on their relevance to the given user) and return the first few items on this ranked list. This task is called the topN recommendation. The result of rating prediction can be transformed to topN recommendations by recommending items with the highest predicted ratings for a given user. Good rating prediction not necessarily translates to good topN recommendation.

Implicit feedback is collected via monitoring the behavior of users while they use a service (e.g. a web shop). User interaction is not required in order to get the feedback, therefore it is available in large quantity. This is of key importance in practical scenarios. However preferences must be inferred from the interactions. The presence of an user action on an item is considered to be a noisy sign of positive preference. It is even harder to infer negative feedback as the absence of an event can be traced back to multiple causes, the most common being that the user does not know about the item. Algorithms working with the implicit problem should consider the "missing" feedback in some way.

Context-awareness: Context-aware recommender systems (CARS)[1] consider additional information (termed contextual information or briefly context) besides user–item interactions. The hypothesis of context-aware recommendations is that they can significantly improve recommendation accuracy, because: (1) Context related effects can be handled during training. For example, certain shifts in the behavior, like seasonal changes, are only understandable with the proper context provided. For algorithms that do not consider context, these variations seem to be semi-random and can not be handled properly, thus the result will be similar to learning on noisy data. (2) Recommendation lists can be tailored according to the actual value of the context, which may influence the users' needs.

Evaluation of recommender algorithms: The offline evaluation of topN recommenders w.r.t. recommendation accuracy is as follows. For a given user–context configuration setting all items are ranked by their predicted preference ($\hat{r}$). Evaluation metrics are calculated on a test set that does not take part in the training in any form. The relevant items for a user–context configuration (i.e. query) are defined as the items on which the user has events under the given context in the test set. Recommended items are the first N items taken from the ranked list of items generated for the query. Generally, I use $N = 20$; results with $N = 10$ and $N = 5$ are usually correlate. It is important to note that I rank all items during evaluation.

There are several metrics for measuring recommendation accuracy. I mostly use recall@N that is the ratio of recommended and relevant items to the relevant items. In other words, it is the proportion of test events that were ranked above in the first N places of their corresponding recommendation list. Recall is a good proxy for certain recommendation settings and usually correlates well with click-through rate (CTR), a commonly used online evaluation metric.

2 Research outline

This work focuses on solving the context-aware implicit feedback based recommendation task with factorization and is heavily influenced by the practical considerations. The aim of the research is to integrate context and eventually other types of information (e.g. metadata) into factorization algorithms in order to increase recommendation accuracy for implicit feedback based topN recommendations. Context is defined as in event context (associated with the transactions, not with the entities of the transaction). The main metric for recommendation accuracy is recall@20.

I use two types of context, *seasonality* and *sequentiality* throughout this work. The practical importance of these context dimensions is that they can be easily created for almost all implicit dataset. They only require that timestamps are associated with the transactions, that is very common in practice.

Seasonality: Many application areas of recommender systems exhibit the seasonality effect, because periodicity can be observed in many human activities. Therefore seasonal data is an obvious choice for context [8]. First we have to define the length of the season. Within a season we do not expect repetitions in the aggregated behavior of users, but we expect that at the same time offset in different seasons, the aggregated behavior of the users will be similar. The length of the season depends on the data. Once we have this, we need to create *time bands* (bins) within seasons that are the possible context-states. Time bands specify the time resolution of a season, which is also data dependent. In the final step, events are assigned to time bands according to their time stamp.

Sequentiality: In some domains, like movies or music, users consume similar items. In other domains, like electronic gadgets or e-commerce in general, they avoid items similar to what they already consumed and look for complementary products. Sequential patterns can be observed in both domain types. Sequentiality as a context dimension was introduced by me in [P2] and uses the previously consumed item by the user as a context for the actual item. This information helps in the characterizations of repetitiveness related usage patterns and sequential consumption behavior.

The evaluation of methods and algorithms is done on five genuine implicit feedback datasets. Three of these are public, two are proprietary.

2.1 Injecting information into matrix factorization through initialization

The first step of this research is to examine if additional information can be used in pure CF factorization methods to increase recommendation accuracy without having to modify the base algorithm. The first step of matrix factorization algorithms is to initialize the feature matrices with random values which will be modified by the training procedure. The idea is to start the factorization from a more sensible starting point, i.e. initialize the item and/or the user feature vectors using information other than the transactions. This way different kinds of information can be injected into the model. Although this does not result in a context-aware solution, it can increase accuracy.

Three main initialization processes are examined. The initialization of feature vectors builds on the observation that feature vectors of similar items are similar. However similarity can be defined in different ways and the additional information can be used for it as well. All three methods start by defining a descriptor matrix for the items (or users) by using the additional information (e.g. metadata or context). Then this descriptor matrix is factorized so we get latent representations for the items and for the entities of the other dimension (metadata terms or context-states). From here there are different possibilities:

- Since the item features of similar items should be similar after the factorization, use the item features from this initial factorization.
- Similarity can be defined between the descriptor vectors, but the computation of the similarity matrix between items is infeasible in practice. But the similarities can be approximated using the latent feature vector. By realizing that the similarity matrix is the product of the descriptor matrix and its transpose, latent feature vectors can be computed for the items – using the feature matrices of the factorized descriptor matrix – so that the similarities between feature vectors approximate the similarity of the descriptor vectors. These feature vectors then can be used for initialization.
- The similarity of items can also be defined as the similarity of how similar they are to other items. While the computation of the exact values is practically infeasible, a method similar to the previous one can be used to compute feature vectors whose relations approximate this similarity.

These methods are used to enhance iALS, a commonly used implicit feedback based matrix factorization method. There is a significant increase in the accuracy compared to random initialization. I also compare metadata and context based initialization and find that context-based generally ranks higher, but their combination can further increase accuracy.

2.2 Context-aware factorization algorithms

In the next part of the research I developed two context-aware algorithms that work efficiently on implicit feedback data. Both algorithms assume the data that is representable in an N_D dimensional tensor R. One dimension of the tensor corresponds to the users (user IDs), one to the items (item IDs), while the other $N_D - 2$ dimension is associated with different context dimensions. R contains only zeroes and ones. Let a given element of the tensor be $r_{u,i,c_1,\cdots,c_{N_D-2}} = 1$ if user u has (at least one) event on item i while the context-state of j^{th} context dimension was c_j. Due to its construction, all elements of R are known (i.e. there are no missing "ratings") however the proportion of ones is very low. This construction of the preference tensor basically assumes that the presence of an event signals positive preference and the absence of an event (i.e. missing feedback) is a sign of negative preference. Since the missing feedback is clearly a weaker signal of negative preference than the presence of positive feedback I construct the $\mathcal{W}(i_1, \ldots, i_{N_D})$ weight function that assigns a real value to every possible entity combination. In practice, the construction of $\mathcal{W}(\cdot)$ depends on the problem, and can also affect the complexity of the training. For the sake of simplicity assume that $\mathcal{W}(\cdot)$ is 1 for missing events and $100 \cdot \#(i_1, \ldots, i_{N_D})$ for non-missing ones. The algorithms optimize for weighted sum of squared errors (equivalent of optimizing for weighted root mean squared error), where the target is R and weights come from $\mathcal{W}(\cdot)$. The algorithms optimize the feature matrices using alternating least squares (ALS). This means that at a given time all but one feature matrices are fixed and the non-fixed one is computed as the least squares solution (given the other, fixed matrices). This process iteratively decreases the value of the loss function. The naive ALS approach on a fully filled tensor (like R) would scale poorly, but careful separation of the computations and precalculation of certain statistics allow the training to be efficient.

The difference between the two algorithms is the preference model, i.e. the expression which is used to compute the predicted preferences.

iTALS: The iTALS algorithm estimates the preferences of user u on item i under the given values of the context dimensions as sum of the values in the Hadamard products (also known as elementwise product) of the corresponding feature vectors. To be less precise, the preference is given by the dot product between the N_D corresponding vectors. This model is referred to as the N-way interaction model (or N-way model for short). The following expression describes the model formally[1]:

$$\hat{r}_{i_1,\dots i_{N_D}} = 1^T \left(M^1_{i_1} \circ M^2_{i_2} \circ \dots \circ M^{N_D}_{i_{N_D}} \right) \tag{1}$$

Generally, this model assumes that all dimensions interact with every other dimension and their interaction results in a preference value. From the recommendation perspective, this model reweights the user–item interaction with a context-configuration dependent feature vector (that is the product of more than one feature vectors if $N_D > 3$).

iTALSx: The method is originally designed to work with three dimensions (users, items and one context). The preference of user u on item i under the given value of the context dimension is predicted as the sum of the dot products between the user and item feature vector, the user and context feature vector and the item and context feature vector. This model is referred to as the pairwise interaction model or pairwise model for short. The model is given by the following expression:

$$\hat{r}_{u,i,c} = 1^T \left(M^{(U)}_u \circ M^{(I)}_i + M^{(U)}_u \circ M^{(C)}_c + M^{(I)}_i \circ M^{(C)}_c \right) \tag{2}$$

In this model the preference is predicted as the composite of a user–item interaction, a context dependent user bias and a context dependent item bias. The context dependent user bias (i.e. user–context interaction) does not take part in the ranking, because recommendations are generated for a given user under a given context-state, thus its value is the same for all items. However it can reduce the effect of context related shifts in the training data, that would be considered noise by a simple matrix factorization.

Complexity: The complexity of one epoch (i.e. computing each matrix once) is $O(N_D N^+ K^2 + \sum_{i=1}^{N_D} S_i K^3)$ for both algorithms, where N_D, $N+$, K are the number of dimensions, events and features and S_i is the size of the i^{th} dimension (i.e. number of items/users/context-states). Thus the algorithms scale linearly with the number of events. Due to the large number of transactions and the growth rate of the set of transactions, this property is very beneficial in practice. The algorithm scales cubically with the number of features in theory. However $N_D N^+ \gg \sum_{i=1}^{N_D} S_i$ and K is small in practice, thus the first term dominates. Therefore the algorithm scales *quadratically* with the number of features in practice.

Comparison: iTALS and iTALSx are compared to (a) a matrix factorization method, (b) a context-aware baseline that is a composite of MF models and (c) to each other; using both seasonality and sequentiality. Key findings are:

[1]Biases are omitted for clearer presentation.

- iTALS and iTALSx both significantly outperform matrix factorization and the context-aware baseline in terms of recommendation accuracy.
- The proposed sequentiality context significantly increases accuracy compared to the context free and to the seasonality based solutions.
- The learning capabilities of iTALS are higher, however its model is more susceptible to noise and to the blurring effect of low factor models. Thus iTALS can outperform iTALSx when the number of features is sufficiently large or if the dataset is denser. These results imply that one should use iTALSx when the dataset is sparse and we can not afford high feature models.

2.3 Speeding-up ALS-based factorization

The training time of the algorithms is key aspect for practical applicability. Faster training allows to (1) capture a more recent state of the system modeled (advantageous for any system, but required for ones where the lifetime of the items is short or new items appear constantly); (2) retrain the models more frequently; (3) apply trade-off between running times and accuracy by using more features or running more epochs. I propose two approximate methods that significantly speed up ALS-learning, especially if the number of features is high, that is, the gain in speed increases as the number of features increases.

The bottleneck of computations in ALS is solving a system of linear equations of size $K \times K$. The proposed methods avoid to directly solve this system.

ALS-CD: The first method uses coordinate descent. Here all but one model parameter is fixed at a given time and a single feature value is computed at once instead of a vector. This reduces the matrix inversion in solving the system to a simple division. The application of this strategy for the implicit setting is not straightforward, because negative examples (missing) events have to be considered as well. I overcome this issue by compressing the missing events into $K+1$ examples.

ALS-CD does not approximate the ALS solution, but gives similar results. The complexity of iTALS/iTALSx using ALS-CD is $O(N_D K^3 + N_D N^+ N_I K + \sum_{i=1}^{N_D} S_i K^2)$ – where N_I is the number of inner iterations – which is linear in K for the range of practically used values, because $N_D N^+ N_I$ dominates N_D and $\sum_{i=1}^{N_D} S_i$.

ALS-CG: The second method uses the conjugate gradient method to approximate the solution of the system of equations. The efficiency of this solver relies on the efficiency of a matrix–vector multiplication between the coefficient matrix and a vector. The coefficient matrix in this case is the sum of a precomputed matrix and a dyadic sum. Therefore the multiplication can be done very efficiently.

If the number of inner iterations equals to K ALS-CG gives the exact same result as ALS. However good approximations can be achieved by using significantly less iterations. The complexity of iTALS/iTALSx using ALS-CG is $O(N_D N^+ N_I K + N_I K^2 \sum_{i=1}^{N_D} S_i)$ which is linear in K for the range of practically used values.

Comparison: Compared to ALS the speed-up is significant. The speed-up factor is ~ 10.6 for CG and ~ 2.9 for CD if $K = 200$ (and it becomes even greater for larger K values). For the more commonly used $K = 80$, the speed-up is ~ 3.5 and ~ 1.3 for CG and CD, respectively. The accuracy results are very similar for the three methods and the deviations are insignificant most of the times. This means that the proposed speed-ups can be used without sacrificing the accuracy.

Compared to one another, CG has more advantageous properties: it is faster, more stable, slightly more accurate and its direct approximation of ALS is beneficial.

2.4 GFF & preference modeling with context

As we have seen with iTALS and iTALSx, different preference models are appropriate for different situations. Certain parameters of the factorization (e.g. number of features) and the dataset (e.g. sparsity) are beneficial for one or the other model. Most factorization methods only use one of these two models (N-way, pairwise), although the number of possible models grows exponentially as the number of dimensions increases. It is also interesting to observe that both of these models are symmetrical, i.e. all dimensions fill the same role; meanwhile there are two distinguished dimensions in every recommendation task, the user and the item. The preference model has an effect on the learning procedure. It is especially problematic if transformations and separation of the computations are required in order to maintain low complexity. And this is exactly the case with the implicit feedback problem. For example iTALS and iTALSx seem very similar, but there are crucial steps that are different and even rely on different precomputed statistics.

The lack of proper exploration of preference modeling is due to the lack of flexible tools in which one can experiment with various models without being required to implement a specific algorithm for each model. I therefore created the General Factorization Framework (GFF), a single, flexible algorithm that takes the preference model as an input and computes latent feature matrices for the input dimensions. GFF allows us to easily experiment with various linear models on any context-aware recommendation task, be it explicit or implicit feedback based. GFF opens up a new research path in preference modeling under context.

The following properties were important at the design of GFF.

- No restriction on context: GFF works on any context-aware recommendation problem independently of the number and the meaning of context dimensions.
- Large preference model class: the only restriction on the preference model is that it must be linear in the dimensions of the problem[2]. This intuitive restriction does not restrict the applicability to real-world problems.
- Data type independence: besides the practically more useful implicit case, explicit problems can be also addressed by simply changing the weighting scheme in the loss function.
- Flexibility: the weighting scheme of GFF is very flexible, enabling to incorporate extra knowledge through the weights such time decay, dwell time dependent weighting, missing not at random hypotheses and more.
- Scalability: GFF scales well both in terms of the number of interactions in the training set and in the number of features. This makes it applicable in real life recommender systems.

GFF allows for experimentation with novel preference models. Using a 4 dimensional context-aware setting (users, items, seasonality, sequentiality) I defined the following components from which models can be assembled:

- ***UI*:** Interaction between users and items, the classic CF model.

[2] Meaning that a dimension can not directly interact with itself in the model

- ***USI*, *UQI*, *USQI*:** The context value dependent reweighting of the user–item relation, i.e. the context influences how the users interact with items. More context dimensions can be used for reweighting. But the more we use, the more sensitive it becomes to noise and more latent features are required for filtering this out [P4].
- ***US*, *UQ*:** The user–context interaction produces a context dependent user bias that does not play role during the ranking but has noise filtering properties during training. We allow only one context in these interactions, because additional contexts would assume that different context dimensions interact somehow.
- ***IS*, *IQ*:** The item–context interaction results in a context dependent item bias that helps in ranking as well as in learning. Only one context is allowed in these interactions.
- ***SQ*:** Interactions between the two context dimensions. Required for the traditional pairwise model.

From these parts I created models that consider certain aspects of the recommendation task. The examined models are:

- **Interaction model ($UI + USI + UQI$):** This model is the composite of the base behavior of the users (UI) and their context-influenced modification of this behavior (USI and UQI). This model assumes that the preferences of the users can be divided into context independent and dependent parts. In the latter the user–item relation is reweighted by a context dependent weight vector. $USQI$ is not included due to the noisiness of reweighting by more than one weight vector simultaneously.
- **Context interaction model ($USI+UQI$):** Preferences in this model are modeled by solely context dependent parts, i.e. it assumes that user–item interactions strongly depend on the context and this dependency affects the whole interaction rather than solely the items or users.
- **Reduced pairwise model ($UI+US+IS+UQ+IQ$):** This model is a minor variation of the traditional pairwise model with the exclusion of the interaction between context dimensions (SQ). The interaction with context is done separately by users and items, i.e. it does not affect the whole user–item relation.
- **User bias model ($UI + US + UQ$):** Here it is assumed that only the user interacts with the other dimensions. This results in a model where the user–item relation is supported by context dependent user biases. Note that during recommendation the user biases are constant, thus do not affect the ranking. However they might filter out some context related noise during training.
- **Item bias model ($UI+IS+IQ$):** This model assumes that the effect of context can be described by context dependent item biases (e.g. items are popular under certain conditions). The item biases affect the ranking as well as filter context related noise during training.
- **A complex model ($UI + US + IS + UQ + IQ + USI + UQI$):** This model is the composite of the reduced pairwise and the interaction model. It can be also treated as a reduced 3-way interaction model from which the context-context interactions are omitted.

Experimental results indicate that these novel models are fit for the task more than the traditional ones as their accuracy is generally higher. The interaction model performs the best (closely followed by the context interaction model). These two models are also intuitively fit the task well. Note that the number of features affects the ranking of the models. Lower number of features is beneficial to models with low order interactions, while models with higher order of interactions work better if the number of features is high. The interaction model works really well in the range of practically used values for the number of features.

3 Organization of the dissertation

The first chapter gives a high level overview of the field recommender systems, a more specific one about the area of implicit feedback based context-aware factorization, specifies the research problem and defines the general setup for experimentation. The second chapter reviews related literature.

The following four chapters (from chapter three to six) cover the bulk of my research in the area and describe my theses:

- Chapter 3 investigates the initialization of matrix factorization and presents two methods that can be used to inject additional information (e.g. item metadata, context) into matrix factorization without modifying the factorization algorithm. The methods and findings form the first thesis group.
- Chapter 4 covers my context-aware factorization algorithms for implicit feedback data. Both the iTALS (second thesis group) and iTALSx algorithms (third thesis group) solve the task in a scalable way. Their comparison and the determination of their appropriateness for a given problem is summed up in the fourth thesis group. This chapter also introduces the sequentiality context dimension that is easy to derive for almost all implicit feedback datasets and its usage can significantly increase the recommendation accuracy.
- Chapter 5 focuses on improving the speed and the scalability w.r.t. the number of features of the ALS training in factorization methods – such are the aforementioned algorithms – without significant decrease in the recommendation accuracy. Two speed-up methods are proposed and examined, one is based on coordinate descent, the other is on conjugate gradient. The two methods are compared to each other and to the naive ALS. The methods and the observations of the experiments form the fifth thesis group.
- Chapter 6 proposes the General Factorization Framework (GFF), a single algorithm that allows for efficient experimentation with different preference models in the implicit feedback based context-aware setting. Using GFF, several novel preference models are proposed that consider the specialties of the context-aware recommendation task. These models are compared to traditional preference models in the area, like the N-way and the pairwise models. GFF, the novel models and the experiments form the sixth thesis group.

The last three chapters of the dissertation sum the work, discuss the application of the results and overview several paths of possible future research.

4 Summary of the new results

Thesis Group 1: I proposed initializing matrix factorization using information on the items (or users) to increase recommendation accuracy. (See Chapter **??** for details. The methods and the results were published in [**?** P3].)

Thesis 1.1 *I proposed to initialize the feature matrices of matrix factorization methods based on the similarities of its entities instead of starting from randomly initialized matrices. The initialization scheme is generic and thus can be applied to any matrix factorization. It consists of two steps: (1) descriptor vectors are assigned to the entities; (2) the descriptors are compressed to fit the size of the feature vectors. I applied the scheme on implicit ALS and showed on five datasets that this type of initialization can increase the recommendation accuracy measured by recall and MAP.*

Thesis 1.2 *I proposed the SimFactor algorithm that yields feature vectors, which preserve the original similarities between entities more accurately. SimFactor does not require the computation of the similarity matrix (which would be infeasible). I showed on five datasets that similarities are better estimated with this algorithms as with pure compression of the descriptor vectors. I also showed that feature vectors yielded by SimFactor are generally better for initializations than those produced by pure compression.*

Thesis 1.3 *I proposed the* Sim^2*Factor algorithm that is able to yield feature vectors whose similarity approximates the similarity between entities, based on how similar they are to the rest of the entities.* Sim^2*Factor does not require the computation of the similarity matrix. I showed that feature vectors of this kind are useful for initialization.*

Thesis 1.4 *I proposed to use context for describing entities. I showed that context based descriptors are better for initialization than metadata based ones. I also showed that the weighted combination of context and metadata based initializations can further improve the recommendation accuracy.*

Thesis Group 2: I proposed the iTALS algorithm to solve the implicit feedback based context-aware recommendation task. (See Section **??** of Chapter **??** for details. The method and the results were published in [**?**].)

> **Thesis 2.1** *I developed iTALS, a tensor factorization method that uses pointwise ranking via optimizing for weighted sum of squared errors. It estimates preferences using the N-way interaction model, i.e. the sum of elements in the elementwise product of feature vectors from each dimension. I showed that iTALS can be applied to solve the implicit feedback based context-aware recommendation problem by using ones and zeroes for positive and missing feedback respectively with higher weights for positive feedback.*
>
> **Thesis 2.2** *I showed that iTALS significantly outperforms the non context-aware implicit matrix factorization and the prefiltering based context-aware baseline with respect to recommendation accuracy, measured by recall.*
>
> **Thesis 2.3** *I demonstrated that iTALS can be trained efficiently on the implicit feedback based context-aware recommendation problem, using alternating least squares. I showed that iTALS can be efficiently used in practice as it scales linearly with the number of events and quadratically with the number of features in the range of practically useful number of feature values.*

Thesis Group 3: I proposed the iTALSx algorithm an alternative solution to the implicit feedback based context-aware recommendation task. (See Section **??** of Chapter **??** for details. The method and the results were published in [P4**?**].)

> **Thesis 3.1** *I developed iTALSx, a tensor factorization method that uses pointwise ranking via optimizing for weighted sum of squared errors. It estimates preferences using the pairwise interaction model, i.e. the sum of dot products between feature vectors from each pair of dimensions. I showed that iTALSx can be applied to solve the implicit feedback based context-aware recommendation problem by using ones and zeroes for positive and missing feedback respectively with higher weights for positive feedback.*
>
> **Thesis 3.2** *I showed that iTALSx significantly outperforms the non context-aware implicit matrix factorization and the prefiltering based context-aware baseline with respect to recommendation accuracy, measured by recall.*
>
> **Thesis 3.3** *I demonstrated that iTALSx can be trained efficiently on the implicit feedback based context-aware recommendation problem, using alternating least squares. I showed that iTALSx can be efficiently used in practice as it scales linearly with the number of events and quadratically with the number of features in the range of practically useful number of feature values.*

Thesis Group 4: I experimented with the iTALS and iTALSx algorithms, compared them and identified easily accessible contexts. (See Section **??** Chapter **??** for details. The results were published in [**?** P4**?**].)

> **Thesis 4.1** *I proposed to use sequentiality as context for recommendations. Sequentiality is the item with which the user previously interacted, before the current one. I argued that this context information is available with every dataset where transactions can be ordered based on their time of occurrence, which is common in practice. I showed that using this information can significantly increase recommendation accuracy to using no context and even to using seasonality as the context in a wide variety of settings (dataset, algorithms, models, number of features).*
>
> **Thesis 4.2** *I compared the strengths and weaknesses of iTALS (N-way model) and iTALSx (pairwise model). I found that the N-way model is more suitable when the number of features is high and/or if the dataset is more dense; and the pairwise model is better otherwise.*

Thesis Group 5: I proposed ways to speed-up ALS learning through using approximate methods. (See Chapter **??** for details. The methods and the results were published in [P6].)

> **Thesis 5.1** *I proposed a general, conjugate gradient based approximation for ALS in ALS based factorization algorithms. I showed that this approximation scales linearly with the number of features in the range of practically used number of feature values. I showed that this allows the usage of higher factor models and finding better trade-offs between running time and accuracy. I showed that the recommendation accuracy is affected only in a minor way if the approximation is used instead of the exact ALS.*
>
> **Thesis 5.2** *I proposed a general, coordinate descent based approximation for ALS in ALS based factorization algorithms. I showed that this approximation scales linearly with the number of features in the range of practically used number of feature values. I showed that this allows the usage of higher factor models and finding better trade-offs between running time and accuracy. I showed that the recommendation accuracy is affected only in a minor way if the approximation is used instead of the exact ALS.*
>
> **Thesis 5.3** *I compared the conjugate gradient and coordinate descent based approximate solutions from a wide variety of aspects. I showed that the conjugate gradient based method is better, because it (a) follows the exact solution more closely in terms of recommendation accuracy; (b) is faster; (c) scales better; and (d) more stable.*
>
> **Thesis 5.4** *I determined a good trade-off between running time and recommendation accuracy for both approximate methods. I proposed to set the number of inner iterations to 2 in order to get this trade-off.*

Thesis Group 6: I proposed a flexible algorithm by the name of GFF to allow for experimentation with novel preference models. (See Chapter **??** for details. The method and the results were published in [**?**].)

Thesis 6.1 *I developed GFF (General Factorization Framework), a single, flexible factorization algorithms for the implicit feedback based context-aware recommendation problem. The flexibility of GFF lies in taking the preference model as an input. The model can use arbitrary number of dimensions and allows using any linear interaction between the subsets of aforementioned dimensions. I demonstrated that this flexibility allows for experimenting with novel preference models. The data model of the basic GFF is the single attribute MDM, which is appropriate for the context-aware problem in practice.*

Thesis 6.2 *I proposed several novel preference models for the context-aware recommendation task. I measured the usefulness of these models w.r.t. recommendation accuracy (measured by recall) on a four dimensional context-aware problem. The context dimensions I used in this problem can be generally derived from all practical datasets based on the timestamp of the events, making them especially important. I showed that there are multiple novel models that outperform traditional models used in the literature.*

Thesis 6.3 *I showed that one of the proposed models, the interaction model generally performs well. This model is the composite of the user–item (UI) and the context reweighted user–item (UCI) relations. It was the best on four datasets out of five datasets and second on the fifth one. The best model on the fifth dataset is the context interaction model that is closely related to the interaction model.*

Thesis 6.4 *I compared the recommendation accuracy of the best novel models in GFF to that of the state-of-the-art factorization methods. The novel models in GFF significantly outperformed the state-of-the-art on three out of five datasets and gave similar results on one.*

Thesis 6.5 *I extended GFF to be compliant with the Multidimensional Dataspace Model and to be able to incorporate additional information, e.g. session data and item metadata more efficiently. The preliminary experiments I executed showed that using session information can significantly increase recommendation accuracy.*

5 Application of the results

My research is heavily influenced by practical considerations.

- The first thesis group presents a relatively cheap way to inject additional information into existing matrix factorization methods through initialization and thus increase their accuracy.
- The second, third and fourth thesis groups focus on context-aware factorization method in the realistic setting of implicit feedback data. Both iTALS and iTALSx scale well with the number of transactions in this setting. The context dimensions used here are also of practical considerations as they can be derived for any implicit dataset that has timestamp associated with its events. The proposed sequentiality context performs really well, as its usage can significantly increase recommendation accuracy.
- The fifth thesis group focuses on improving the speed and scalability (w.r.t. the number of features) of the ALS training in factorizations without sacrificing its accuracy. This is really important in practice, because lower training times allow for more frequent retraining, finding better trade-offs between training time and accuracy (e.g. by using more features, running more epochs, etc.) and using computational resources more efficiently.
- The sixth thesis group focuses on an algorithm that allows for flexible experimentations with different context-aware preference models. The need for such an algorithm also originates from practice as traditional context-aware models do not consider the specialties of the recommendation task (e.g. distinction of dimensions, proper interactions between dimensions, etc.).

The algorithms and the know-how resulting from this work have been successfully applied in practice. Some of the algorithms are implemented in the recommendation engine of Gravity Research & Development Inc., a recommendation service providing company with clients from all around the world in different domains. The algorithms were used successfully in the live system as well as in other recommendation projects, tenders and POCs. The domains of the application include but not limited to online grocery shopping, VoD and live program recommendation on IPTV [P8], e-commerce web shops and classified sites.

The results also greatly contribute to a project of the European Union's Seventh Framework Programme (FP7/2007-2013) by the name of CrowdRec[3]. CrowdRec aims for creating the next generation of (practical) recommender systems by using context, interactions with the users, analyzing streams and information from heterogeneous sources. My work falls into the context related part of CrowdRec.

[3]Grant Agreement n°610594

6 List of publications

[P1] Balázs Hidasi and Domonkos Tikk. Enhancing matrix factorization through initialization for implicit feedback databases. In *Proceedings of the 2Nd Workshop on Context-awareness in Retrieval and Recommendation*, CaRR '12, pages 2–9, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1192-2. doi: 10.1145/2162102.2162104. URL `http://doi.acm.org/10.1145/2162102.2162104`.

[P2] Balázs Hidasi and Domonkos Tikk. Fast als-based tensor factorization for context-aware recommendation from implicit feedback. In Peter A. Flach, Tijl De Bie, and Nello Cristianini, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 7524 of *Lecture Notes in Computer Science*, pages 67–82. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-33485-6. doi: 10.1007/978-3-642-33486-3_5. URL `http://dx.doi.org/10.1007/978-3-642-33486-3_5`.

[P3] Balázs Hidasi and Domonkos Tikk. Initializing matrix factorization methods on implicit feedback databases. *Journal of Universal Computer Science*, 19(12): 1834–1853, jun 2013. ISSN 0948-695x. URL `http://www.jucs.org/jucs_19_12/initializing_matrix_factorization_methods`.

[P4] Balázs Hidasi. Factorization models for context-aware recommendations. *Infocommunications Journal*, VI(4):27–34, 2014. ISSN 2061-2079.

[P5] Balázs Hidasi and Domonkos Tikk. General factorization framework for context-aware recommendations. *Data Mining and Knowledge Discovery*, pages 1–30, 2015. ISSN 1384-5810. doi: 10.1007/s10618-015-0417-y. URL `http://dx.doi.org/10.1007/s10618-015-0417-y`.

[P6] Balázs Hidasi and Domonkos Tikk. Speeding up ALS learning via approximate methods for context-aware recommendations. *Knowledge and Information Systems*, pages 1–25, 2015. ISSN 0219-1377. doi: 10.1007/s10115-015-0863-2. URL `http://dx.doi.org/10.1007/s10115-015-0863-2`.

[P7] Balázs Hidasi. Context-aware preference modeling with factorization. In *Proceedings of the 9th ACM Conference on Recommender Systems*, RecSys '15, pages 371–374, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3692-5. doi: 10.1145/2792838.2796543. URL `http://doi.acm.org/10.1145/2792838.2796543`.

[P8] Dávid Zibriczky, Balázs Hidasi, Zoltán Petres, and Domonkos Tikk. Personalized recommendation of linear content on interactive tv platforms: beating the cold start and noisy implicit user feedback. In *UMAP Workshops. Proceedings of the International Workshop on TV and Multimedia Personalization*, 2012.

[P9] Balázs Hidasi and Domonkos Tikk. Context-aware item-to-item recommendation within the factorization framework. In *Proceedings of the 3rd Workshop on Context-awareness in Retrieval and Recommendation*, CaRR '13, pages 19–25, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-1847-1. doi: 10.1145/2442670.2442675. URL `http://doi.acm.org/10.1145/2442670.2442675`.

[P10] Balázs Hidasi and Domonkos Tikk. Approximate modeling of continuous context in factorization algorithms. In *Proceedings of the 4th Workshop on Context-Awareness in Retrieval and Recommendation*, CARR '14, pages 3–9, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2723-7. doi: 10.1145/2601301.2601303. URL `http://doi.acm.org/10.1145/2601301.2601303`.

[P11] Balázs Hidasi and Csaba Gáspár-Papanek. Shifttree: An interpretable model-based approach for time series classification. In Dimitrios Gunopulos, Thomas Hofmann, Donato Malerba, and Michalis Vazirgiannis, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 6912 of *Lecture Notes in Computer Science*, pages 48–64. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-23782-9. doi: 10.1007/978-3-642-23783-6_4. URL `http://dx.doi.org/10.1007/978-3-642-23783-6_4`.

[P12] Balázs Hidasi. Újfajta, automatikus, döntési fa alapú adatbányászati módszer idősorok osztályozására. Heszberger Zalán (szerk.), *Proceedings of the 2009. évi Végzős Konferencia*. ISBN 978-963-421-588-2.

[P13] Balázs Hidasi. Modell alapú idősor-osztályozó fejlesztése és kiterjesztése. M.Sc. Thesis, Budapest University of Technology and Economics, Department of Telecommunications and Mediainformatics.

[P14] Balázs Hidasi. Újfajta, automatikus, döntési fa alapú adatbányászati módszer idősorok osztályozására. B.Sc. Thesis, Budapest University of Technology and Economics, Department of Telecommunications and Mediainformatics.

[P15] Balázs Hidasi. Az idősor-osztályozás problémájának megoldása új, döntési fa alapú adatbányászati algoritmussal. XXIX. National Students' Scientific Conference (OTDK), Debrecen, 2009.

References

[1] G. Adomavicius and A. Tuzhilin. Context-aware recommender systems. In *Recsys'08: ACM Conf. on Recommender Systems*, pages 335–336, 2008.

[2] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. *Modern information retrieval*, volume 463. ACM press New York, 1999.

[3] R. Bell and Y. Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *ICDM'07: IEEE Int. Conf. on Data Mining*, pages 43–52, 2007.

[4] J. Bennett and S. Lanning. The Netflix Prize. In *KDD Cup Workshop at SIGKDD'07*, pages 3–6, 2007.

[5] Mukund Deshpande and George Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

[6] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *SIGKDD'08: ACM Int. Conf. on Knowledge Discovery and Data Mining*, pages 426–434, 2008.

[7] G. Linden, B. Smith, and J. York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[8] N. N. Liu, B. Caoand M. Zhao, and Q. Yang. Adapting neighborhood and matrix factorization models for context aware recommendation. In *CAMRa'10: Workshop on Context-Aware Movie Recommendation*, pages 7–13, 2010. ISBN 978-1-4503-0258-6.

[9] P. Lops, M. Gemmis, and G. Semeraro. Content-based recommender systems: State of the art and trends. In *Recommender Systems Handbook*, pages 73–105. Springer, 2011.

[10] I. Pilászy and D. Tikk. Recommending new movies: Even a few ratings are more valuable than metadata. In *Recsys'09: ACM Conf. on Recommender Systems*, pages 93–100, 2009. ISBN 978-1-60558-435-5.

[11] I. Pilászy, D. Zibriczky, and D. Tikk. Fast ALS-based matrix factorization for explicit and implicit feedback datasets. In *Recsys'10: ACM Conf. on Recommender Systems*, pages 71–78, 2010. ISBN 978-1-60558-906-0.

[12] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.

[13] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor. *Recommender Systems Handbook*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010. ISBN 0387858199, 9780387858197.

[14] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems 20*. MIT Press, 2008.

[15] J Ben Schafer, Joseph A Konstan, and John Riedl. E-commerce recommendation applications. In *Applications of Data Mining to Electronic Commerce*, pages 115–153. Springer, 2001.

[16] G. Takács, I. Pilászy, B. Németh, and D. Tikk. Major components of the Gravity recommendation system. *SIGKDD Explor. Newsl.*, 9:80–83, December 2007. ISSN 1931-0145.

[17] G. Takács, I. Pilászy, and D. Tikk. Applications of the conjugate gradient method for implicit feedback collaborative filtering. In *RecSys'11: ACM Conf. on Recommender Systems*, pages 297–300, 2011.