Algorithms for time-aware recommender systems

A presentation by Sami Diaf and Carlo Morgenstern

Δt

Δt

Structure

1. Topic and Motivation
2. Paper: Collaborative Filtering with Temporal Dynamics
   1. Objective
   2. Preliminaries
   3. Time-aware factor models
   4. Time changing factor model
   5. Temporal dynamics at neighbourhood models
   6. Conclusion
3. Paper: Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering
   1. Objective
   2. Preliminaries
   3. General Tensor Factorization
   4. Tensor Factorization for Collaborative Filtering
   5. Regularization and Optimization
   6. Pseudocode
   7. Conclusion
4. Comparison of the algorithms

Topic and Motivation

Data is changing over time, thus, there is a constant need to update models in order to reflect its present nature.

Analysis of such data should find the right balance between:

- Discounting temporary effects (having a low impact on future behaviour)
- Long–term trends reflecting the inherent nature of data

Example : seasonal changes (specific holidays) leading to shopping patterns

1 Δt

2

3

4

Topic and Motivation

Item–side effects:

- Product perceptions and popularity are in a constant change.
- Seasonal patterns influence item’s popularity.

User–side effects:

- Customers redefine their tastes.
- Transient, short–term bias, anchoring
- Drifting rating scale
- Change of rater within the household

1 Δt

2

3

4

Collaborative Filtering with Temporal Dynamics

By Yehuda Koren

Objective

The primary objective of the paper is to model user preferences for building a recommender system.

Techniques of addressing time–changing user preferences will serve to build two recommender techniques:

- Factor modelling
- Item–item neighbourhood

Objective

Recommender systems are often based on collaborative filtering (CF), a technique relying on past user behaviour (previous ratings, previous transactions...) to analyse relationships between users and interdependencies among products, in order to identify new user–item associations.

Two primary areas of CF:

– Latent factor models → alternative approach trying to explain the ratings by characterizing both items and users (matrix factorization is the most successful example

– Neighbourhood methods → computing relationship between items or, alternatively, between users

Objective

Basic matrix factorization model

users

items

1		3			5			5		4	
		5	4			4			2	1	3
2	4		1	2		3		4	3	5	
	2	4		5			4			2	
		4	3	4	2					2	5
1		3		3			2			4	

~

items

.1	-.4	.2
-.5	.6	.5
-.2	.3	.5
1.1	2.1	.3
-.7	2.1	-2
-1	.7	.3

●

users

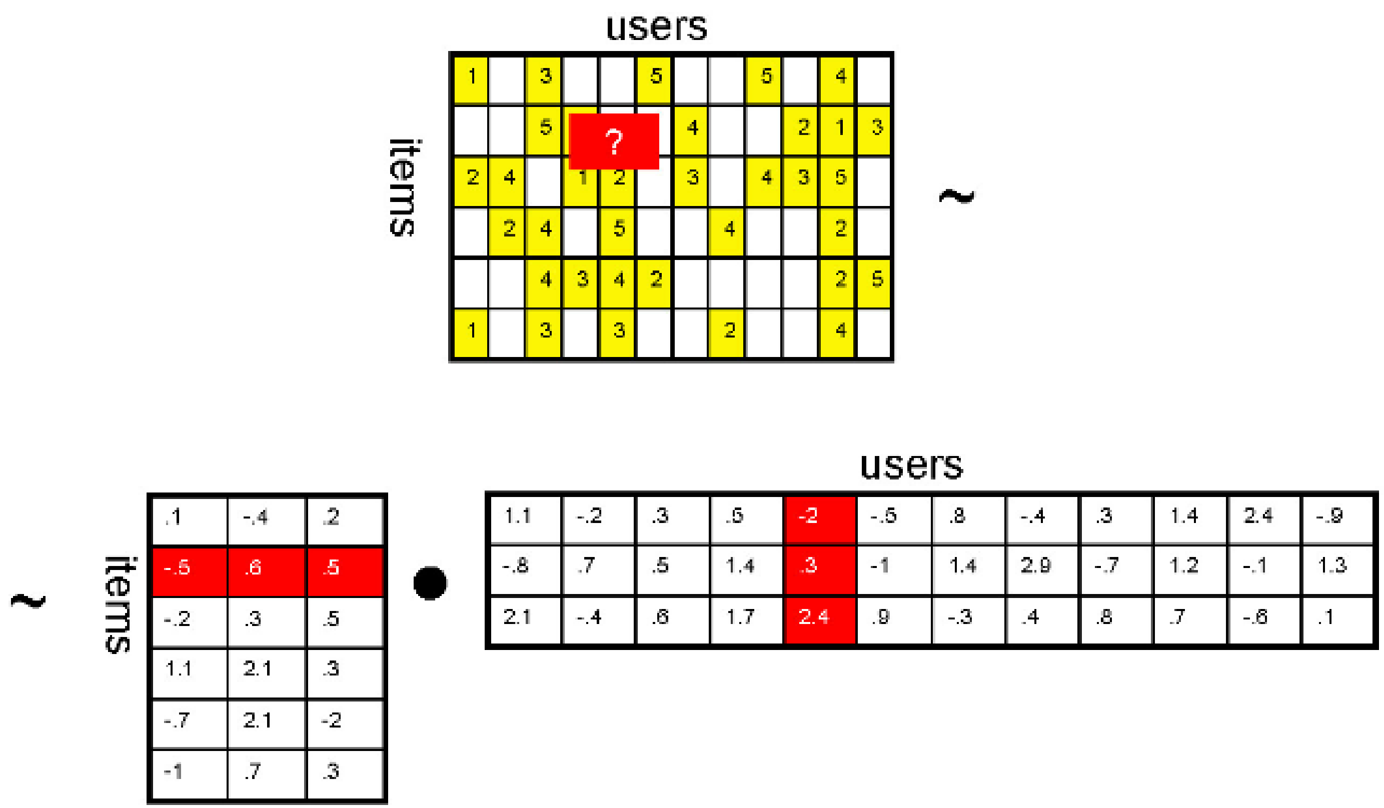
1.1	-.2	.3	.5	-2	-.5	.8	-.4	.3	1.4	2.4	-.9
-.8	.7	.5	1.4	.3	-1	1.4	2.9	-.7	1.2	-.1	1.3
2.1	-.4	.6	1.7	2.4	.9	-.3	.4	.8	.7	-.6	.1

A rank-3 SVD approximation

Y!Research

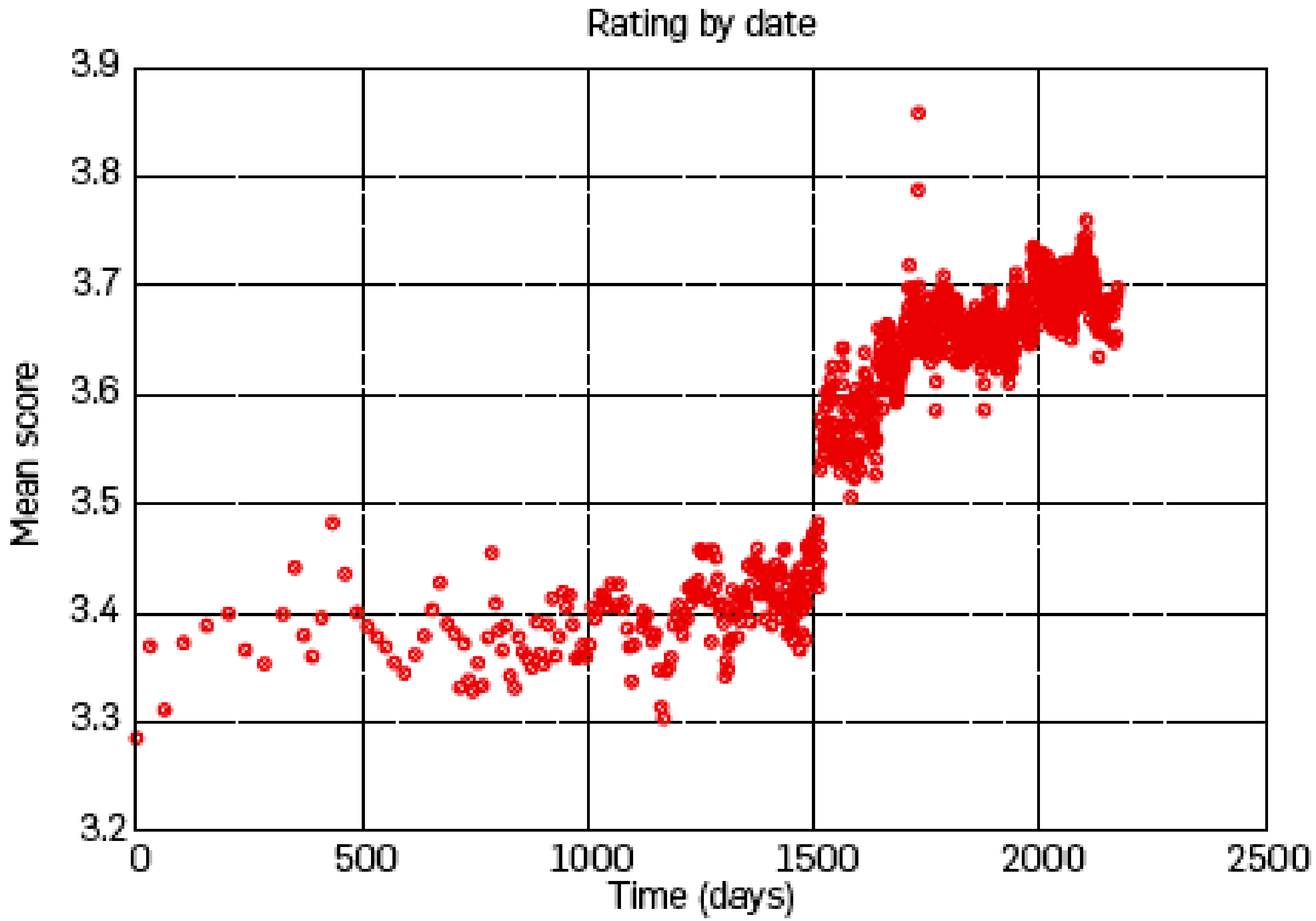
Objective

Estimate unknown ratings as inner-products of factors:

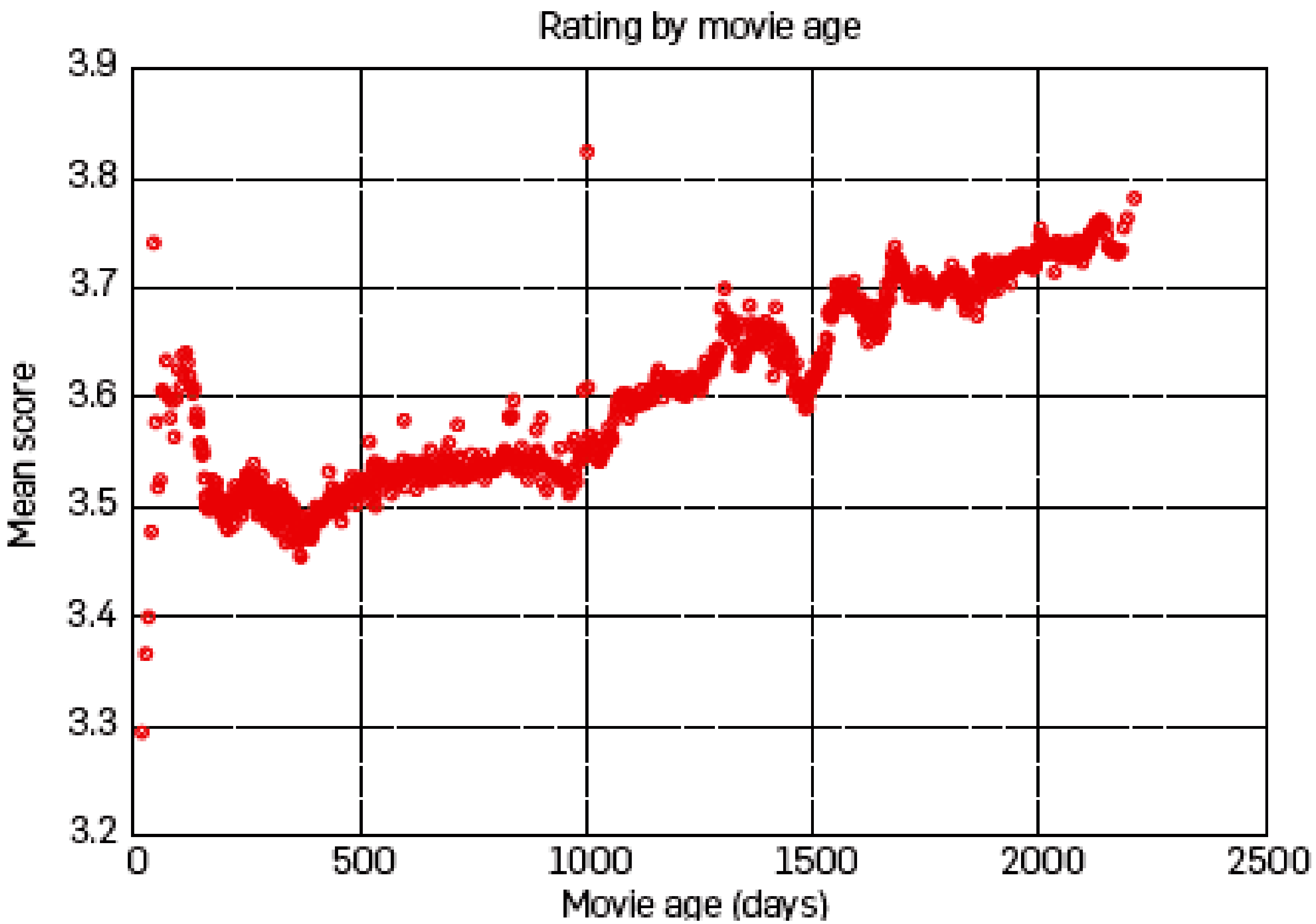


A rank-3 SVD approximation

Objective


Rating by date
Mean score
Time (days)
3.9
3.8
3.7
3.6
3.5
3.4
3.3
3.2
0
500
1000
1500
2000
2500

Objective


Rating by movie age
3.9
3.8
3.7
3.6
3.5
3.4
3.3
3.2
Mean score
0
500
1000
1500
2000
2500
Movie age (days)

Preliminaries - Notation

- Given ratings for m users (customers) and n items (products)
- Special indexing letters:
  - For users: u, v
  - For items: i, j
- r_{ui} indicates the preference by user u of item i
- Scalar t_{ui} denotes the time of the rating r_{ui}
- The training set is $\mathrm{K} = \{(u, i) \mid r_{ui} \; is \; known\}$

Preliminaries - Data

– Over 100 million date-stamped ratings performed by about 480,000 anonymous Netflix customers on 17,770 movies between 31.12.1999 and 31.12.2005.

– Ratings are integers ranging from 1 to 5.

– A movie receives, on average, 5,600 ratings, while a user rates 208 movies.

– For comparability with result two sets:

  – Hold–out set *Probe set*

  – Test set *Quiz set*

– Quality of results measured by the root mean squared error:

$$RMSE = \sqrt{\frac{\sum_{u,i \in K} (r_{ui} - \hat{r}_{ui})^2}{|TestSet|}}$$

Time-aware factor models – static factor model

- Each user u is associated with a vector $p_u \in \Re^f$
- Each item i is associated with a vector $q_i \in \Re^f$
- The rating is predicted by:

$$\hat{r}_{ui} = {q_i}^T p_u = \sum_{k=1}^{f} q_i[k]\, p_u[k] \qquad (1)$$

- The challenge → computing the mapping of each item and user to factor vectors q_i and p_u
- Once the mapping accomplished, we can easily compute the ratings a user will give to any item by using the above equation.

Time-aware factor models – static factor model

- This approach is closely linked to the single value decomposition.
- Due to the data distortion, Koren et al. suggested modeling directly only the observed ratings, while avoiding overfitting through an adequate regularized model.
- To learn q_i and p_u we use the stochastic gradient descent (l2-regularization)

$$min_{q^*,p^*} \sum_{u,i \in K} (r_{ui} - {q_i}^T p_u)^2 + \lambda(||q_i||^2 + ||p_u||^2) \qquad (2)$$

Time-aware factor models – static factor model

– Eq.2 captures interactions between users and items, but much of the observed variation in rating values is due to effects associated with either users or items

$\rightarrow$ building a static baseline predictor b_{ui} for an unknown r_{ui} which includes deviations of user u and item i from the averages:

$$b_{ui} = \mu + b_u + b_i \tag{3}$$

– Example: average rating over all movies $\mu = 3.7$. Titanic tends to be rated 0.5 stars above the average. User Joe is critical and tends to give 0.3 stars below the average. So the baseline estimate $\hat{b}_{ui} = 3.7 - 0{,}3 + 0{,}5 = 3.9$

1 | Δt

2 | 1

2

3

4

5

3 | 6

4

Time-aware factor models – static factor model

– The baseline predictor b_{ui} should be integrated in Eq.1

$$\hat{r}_{ui} = \mu + b_u + b_i + {q_i}^T p_u \quad (4)$$

- – μ the global average
- – b_u the user bias
- – b_i the item bias
- – ${q_i}^T p_u$ user-item interaction

– Learning is done by minimizing the squared error function

$$min_{q^*,p^*} \sum_{u,i \in K} (r_{ui} - \mu - b_u - b_i - {q_i}^T p_u)^2 + \lambda\left(\|q_i\|^2 + \|p_u\|^2 + {b_u}^2 + {b_i}^2\right) \quad (5)$$

1 | Δt
2 | 1 2 3 4 5 6
3
4

Time-aware factor models – movie-related temporal effects

- Bias could be better treated as a function of time, to translate the ability of users to change their baseline ratings over time.

$$b_{ui} = \mu + b_u(t_{ui}) + b_i(t_{ui}) \tag{6}$$

- $b_u(t_{ui})$ and $b_i(t_{ui})$ are real valued functions that change over time, and they will be split into time–based bins

$$b_i(t) = \mu + b_i + b_{i,Bin(t)} \tag{7}$$

- Each bin corresponds to roughly 10 consecutive weeks of data (30 bins for the whole dataset).

1 Δt

2 1 2 3 4 5 6

3

4

Time-aware factor models – linear modelling of user bias

– Binning the parameters works well on the items i but not on users u, thus a linear function was adopted to capture possible gradual shift of user-bias.

$$dev_u(t) = sign(t - t_u) * |t - t_u|^{\beta}$$

with t_u the mean date of rating and β set to 0.4 (by cross–validation).

$$b_u^{(1)}(t) = b_u + \alpha_u * dev_u(t) \qquad (8)$$

1 2 3 4

Δt

1 2 3 4 5 6

Time-aware factor models – linear modelling of user bias + single day effect

– Another parameter designed to absorb the day–specific variability b_{ut}

$$b_{ui} = \mu + b_u + \alpha_u * dev_u(t_{ui}) + b_i + b_{i,Bin(t_{ui})} \qquad (9)$$

– So, the model will be written as:

$$b_{ui} = \mu + b_u + \alpha_u * dev_u(t_{ui}) + b_{u,t_{ui}} + b_i + b_{i,Bin(t_{ui})} \qquad (10)$$

Time-aware factor models - Comparison

Model	Static model	Movie-related	Linear user bias	Linear user bias +
RMSE	0.9799	0.9771	0.9731	0.9605

1

2

3

4

Δt

1

2

3

4

5

6

Time-aware factor models – Capturing periodic effect

– Concerns both items and users

– Some items are more popular in specific seasons or near holidays

$$b_i(t) = \mu + b_i + b_{i,Bin(t)} + b_{i,\,period(t)} \quad (11)$$

– Users may have different buying patterns during weekends, holidays ..., compared to working days.

$$b_u(t) = b_u + \alpha_u * dev_u(t) + b_{u,t} + b_{u,period(t)} \quad (12)$$

– Results showed no significant periodic effects for the Netflix data !

Time-aware factor models – Changing scale of user rating

– Users may employ different rating scales.

– A single user may can change his rating scale over time.

– Thus, a time–dependent scaling feature $c_u(t) = c_u + c_{ut}$ is suggested, where c_u is stable part and c_{ut} represents day–specific variability.

$$b_{ui} = \mu + b_u + \alpha_u * dev_u(t_{ui}) + b_{u,t_{ui}} + \left(b_i + b_{i,Bin(t_{ui})}\right) * c_u(t_{ui}) \quad (13)$$

– Adding $c_u(t)$ to the baseline predictor lowers RMSE to 0.9555 which is close to the Netflix's Cinematch recommender system (RMSE=0.9514)!

Time changing factor model

– Temporal dynamic also affects the interaction between users and items.

– The user factors p_u will be considered as a function of time: $p_u(t) = (p_u(t)[1], \ldots, p_u(t)[f])$

$$p_u(t)[k] = p_{uk} + \alpha_{uk} * dev_u(t) + p_{ukt} k = 1, \ldots, f \quad (14)$$

– The SVD basic factor model: $\hat{r}_{ui} = \mu + b_u + b_i + {q_i}^T p_u$ becomes:

$$\hat{r}_{ui} = \mu + b_u(t_{ui}) + b_i(t_{ui}) + {q_i}^T p_u(t_{ui}) \quad (15)$$

– Learning performed by regularized stochastic gradient descent (convergence not affected by the temporal parametrization)

Time changing factor model – factor models used in practice

- SVD (Singular Value Decomposition)
- SVD++, a simplified version of an asymmetric SVD, which uses explicit ratings (Koren 2008)
- timeSVD++, a simplified version that includes time–varying features

Model	f = 10	f = 20	f = 50	f = 100	f = 200
SVD	0.9140	0.9074	0.9046	0.9025	0.9009
SVD++	0.9131	0.9032	0.8952	0.8924	0.8911
timeSVD++	0.8971	0.8891	0.8824	0.8805	0.8799

Temporal dynamics at neighbourhood models – Static model

- Neighborhood models is the most common approach to Collaborative Filtering
- A static item–item model is:

$$\hat{r}_{ui} = \mu + b_i + b_u + |R(u)|^{-1} \sum_{j \,\epsilon\, R(u)} (r_{ui} - b_{ui}) w_{ij} + c_{ij} \tag{16}$$

  - $|R(u)|$ contains the items rated by the user u
  - w_{ij} is the needed adjustment for the values of the rating
  - c_{ij} disregards the rating value by considering only the which items were rated
  - w_{ij} and c_{ij} are not expected to shift over time

1 Δt

2

1

2

3

4

5

6

3

4

Temporal dynamics at neighbourhood models – Time-varying model

- There is a need to parametrize the decaying relations between two items rated by user u
- Exponential decaying $e^{-\beta_u * \Delta t}$ was used, where $\beta_u > 0$ controls the user-specific decay rate (learned from data)
- $(1+\beta_u \Delta t)^{-1}$ is another alternative to exponential decaying with the same accuracy but faster.

$$\hat{r}_{ui} = \mu + b_i(t_{ui}) + b_u(t_{ui}) + |R(u)|^{-1} \sum_{j \,\epsilon\, R(u)} e^{-\beta_u * |t_{ui} - t_{uj}|} (r_{uj} - b_{uj}) w_{ij} + c_{ij} \tag{17}$$

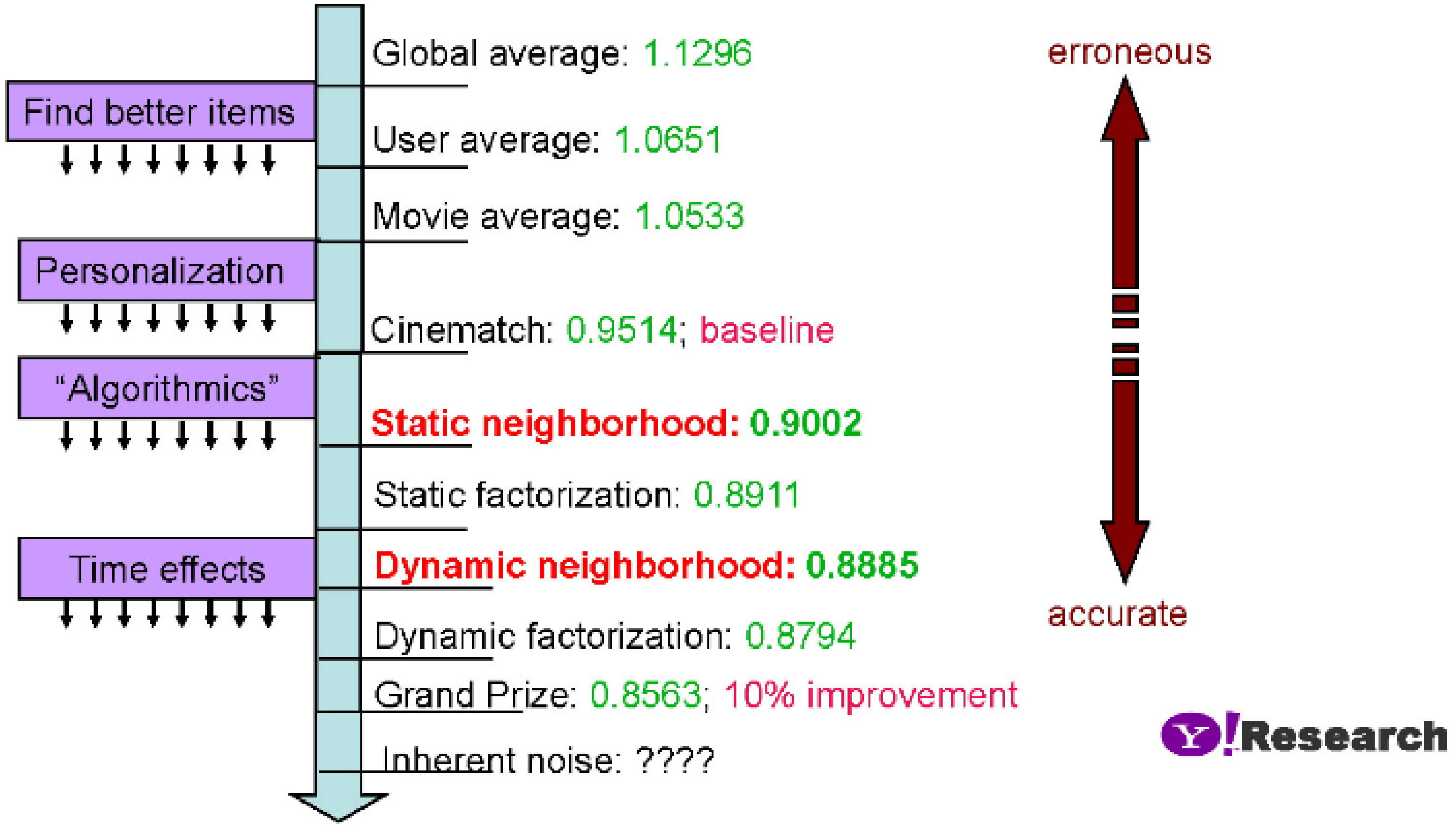
1 2 3 4

Δt 1 2 3 4 5 6

Temporal dynamics at neighbourhood models – Time-varying model

- $|R(u)|$ contains the items rated by the user u
- w_{ij} is the needed adjustment for the values of the rating
- c_{ij} disregards the rating value by considering only the which items were rated
- w_{ij} and c_{ij} are not expected to shift over time
- RMSE decreases from 0.9002 to 0.8885

Temporal dynamics at neighbourhood models – Time-varying model



1 2 3 4

Δt 1 2 3 4 5 6

Conclusion

– Modelling temporal effects significantly improves recommenders accuracy.

– Multiple time drifting patterns across users and items.

– Sudden single–day effects are significant.

– Past temporal fluctuations may help predict future behaviour.

1 Δt

2

Multiverse Recommendation: N-dimensional Tensor Factorization for Context-aware Collaborative Filtering

3

By Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas and Nuria Oliver

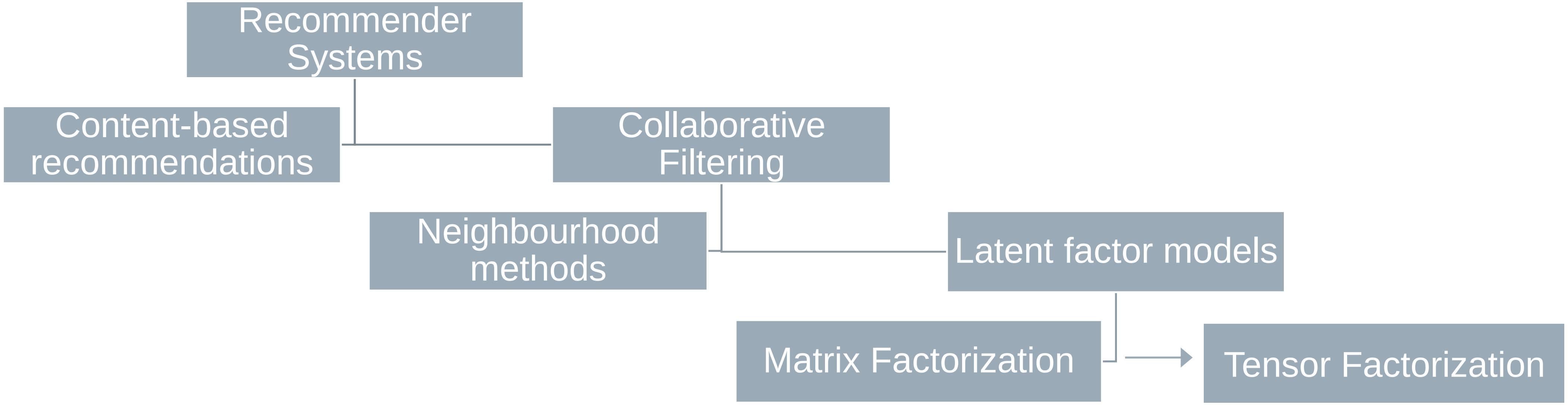
4

Objective

– integrating context into Recommender Systems

– specifically into Matrix Factorization models

– generalization of 2D user-item matrix to nD user-item-context tensor



1 Δt

2 1

2

3 3

4

5

6

7

4

Preliminaries

initial sparse tensor $Y \in \mathcal{Y}^{n\times m\times c}$

n - number of users *n*

u - index of specific user *u*

m - number of items *m*

i - index of specific item *i*

c - number of values of the context variable

k - index of specific context variable

$\times_U$ - tensor-matrix multiplication, where subscript shows which dimension to multiply the matrix with

$$T = Y \times_U U \text{ equals } T_{ljk} = \sum_{i=1}^{n} Y_{ijk} U_{ij}$$

$\otimes$ - outer tensor product

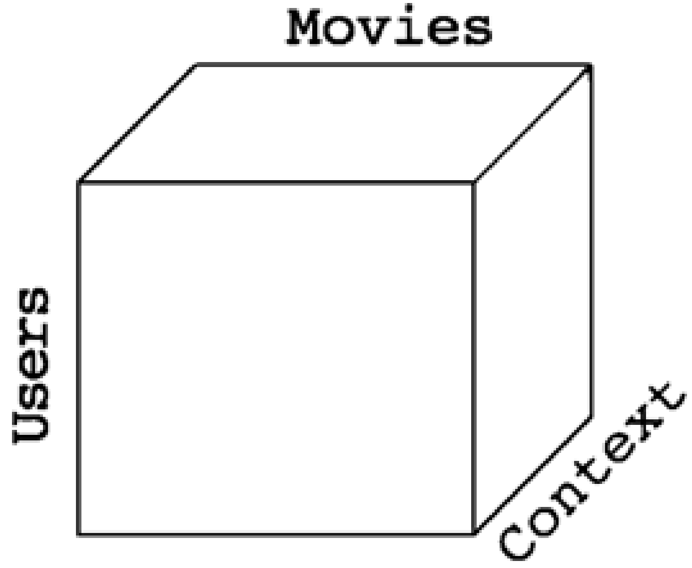


Fig. 1

General Tensor Factorization

- in short: TF
- HOSVD is used

$$Y \in \mathcal{Y}^{n \times m \times c} \rightarrow \mathrm{U} \in \mathbb{R}^{n \times d_U},$$
$$I \in \mathbb{R}^{m \times d_I}, C \in \mathbb{R}^{c \times d_C},$$
$$S \in \mathbb{R}^{d_u \times d_I \times d_C}$$

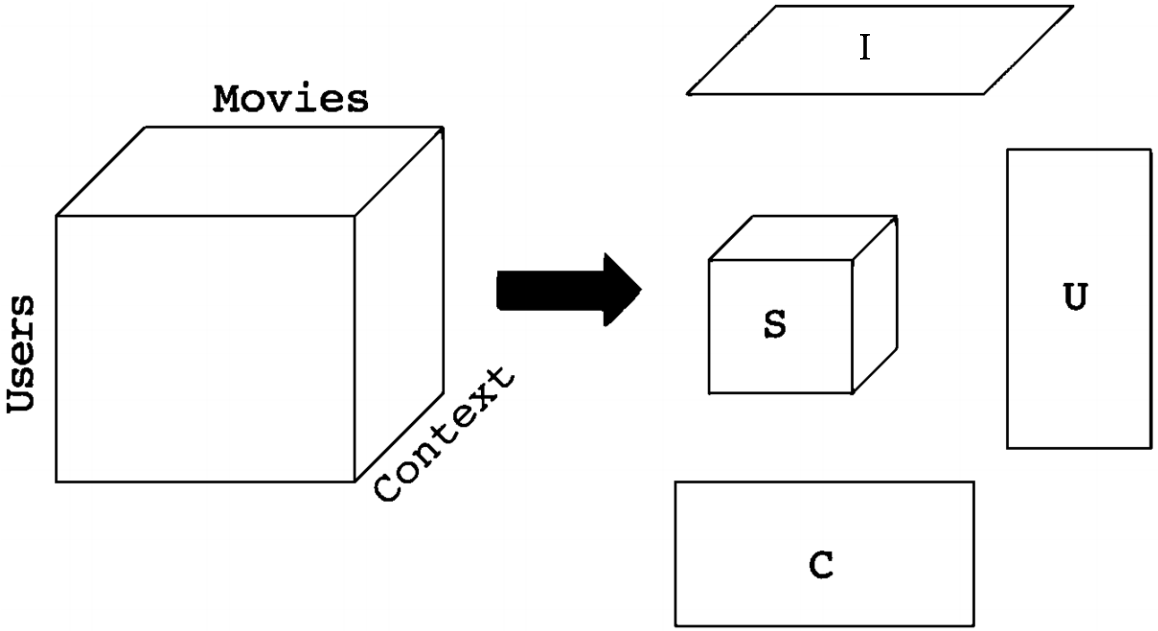


Fig. 2

Decision function for single user, item and context:

$$F_{uik} = S \times_U U_{u*} \times_I I_{i*} \times_C C_{k*} = \hat{R}_{uik}$$

1 | Δt
2 | 1 2 3 4 5 6 7
3
4

Tensor Factorization for Collaborative Filtering

– minimizing loss on observed data

– additional binary tensor D_{uik}, that is 1 on every observed value

$$L(\hat{R}, Y) = \frac{1}{\|S\|_1} \sum_{u,i,k} D_{uik} l(\hat{R}_{uik}, Y_{uik})$$

where l is a pointwise loss function

e.g. $l(f, y) = |f - y|$

1 2 3 4

Δt

1 2 3 4 5 6 7

Regularization and Optimization

– to avoid overfitting of the model

Frobenius norm: $\Omega(U,M,C) = \frac{1}{2}(\lambda_U \|U\|_F^2 + \lambda_I \|I\|_F^2 + \lambda_C \|C\|_F^2)$

$$\Omega(S) = \frac{1}{2}(\lambda_S \|S\|_F^2)$$

Resulting minimization problem (risk functional R):

$$\mathrm{R}(U,M,C,S) = \frac{1}{\|S\|_1} \sum_{u,i,k} D_{uik}\, l(\hat{R}_{uik}, Y_{uik}) + \frac{1}{2}(\lambda_U \|U\|_F^2 + \lambda_I \|I\|_F^2 + \lambda_C \|C\|_F^2) + \frac{1}{2}(\lambda_S \|S\|_F^2)$$

– usage of stochastic gradient descent (SGB) instead of subspace descent for performance reasons

Pseudocode

Input Y, d, λ

Initialize $U \in \mathbb{R}^{n \times d_U}$, $I \in \mathbb{R}^{m \times d_I}$, $C \in \mathbb{R}^{c \times d_C}$, $S \in \mathbb{R}^{d_u \times d_I \times d_C}$ with small random values.

$t = t_0$

for each (u,i,k) in observed values of Y **do**

$$\eta = \frac{1}{\sqrt{t}} \text{ and } t = t + 1$$

$$\hat{R}_{uik} = S \times_U U_{u*} \times_I I_{i*} \times_C C_{k*}$$

$$U_{u*} = U_{u*} - \eta\lambda_U U_{u*} - \eta \partial_{\hat{R}_{uik}} l(\hat{R}_{uik}, Y_{uik}) S \times_I I_{i*} \times_C C_{k*}$$

$$I_{i*} = I_{i*} - \eta\lambda_I I_{i*} - \eta \partial_{\hat{R}_{uik}} l(\hat{R}_{uik}, Y_{uik}) S \times_U U_{u*} \times_C C_{k*}$$

$$C_{k*} = C_{k*} - \eta\lambda_C C_{k*} - \eta \partial_{\hat{R}_{uik}} l(\hat{R}_{uik}, Y_{uik}) S \times_U U_{u*} \times_I I_{i*}$$

$$S = S - \eta\lambda_S S - \eta \partial_{\hat{R}_{uik}} l(\hat{R}_{uik}, Y_{uik}) U_{u*} \otimes I_{i*} \otimes C_{k*}$$

end

Output U, M, C, S

Conclusion

– evaluation on 3 relatively small data sets

Data set	Users	Items	Scale	Ratings	Context Dimensions
Yahoo!	7642	11915	1-5	221k	2 (3 age groups, synthetically created strong context)
Adomavicius	84	192	1-13	1464	5 (companion, weekday, season, year, isFirstWeekend)
Hideki	212	20	1-5	6360	2 (3 “degrees of hunger”, isRealSituation)

– comparison with vanilla matrix factorization

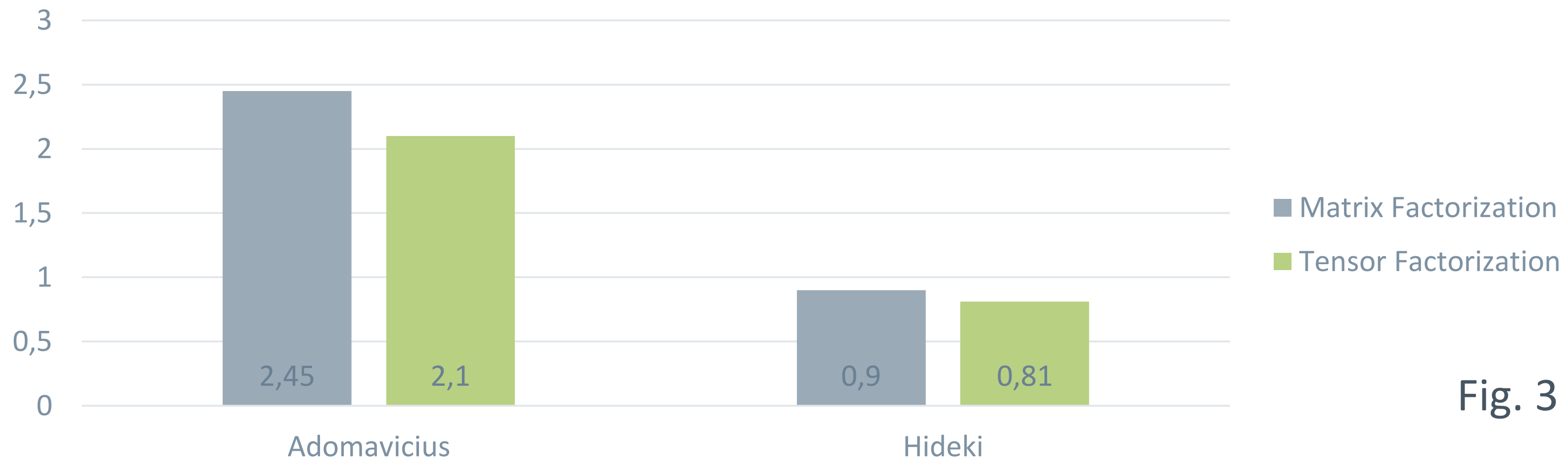
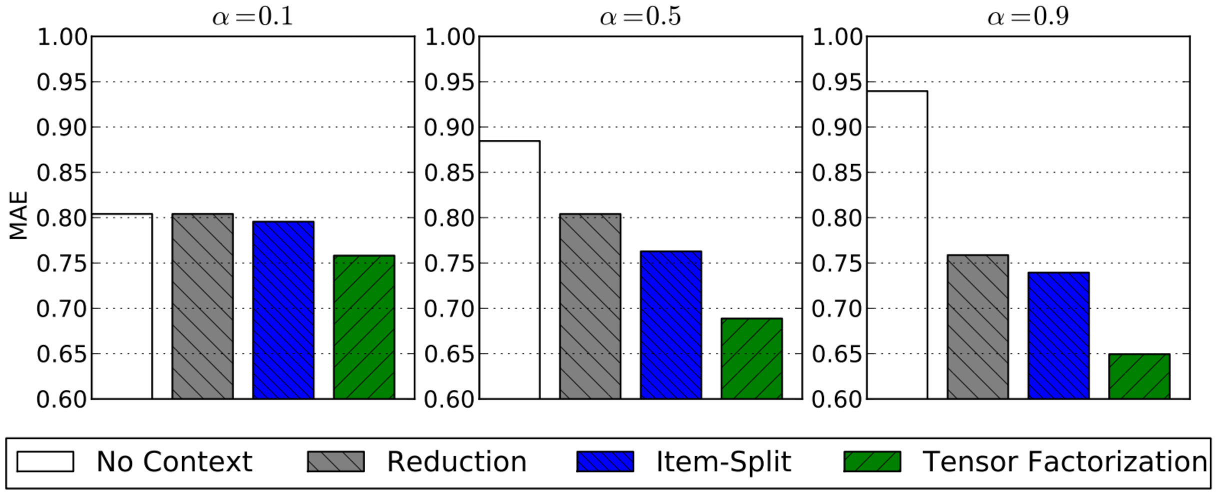


Fig. 3

Conclusion

– comparison with pre-filtering context-aware methods



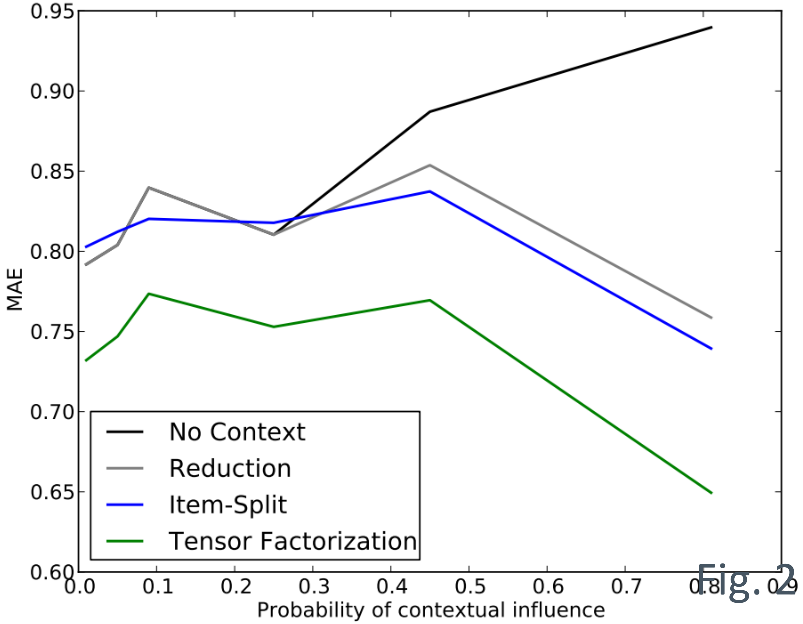


Fig. 2

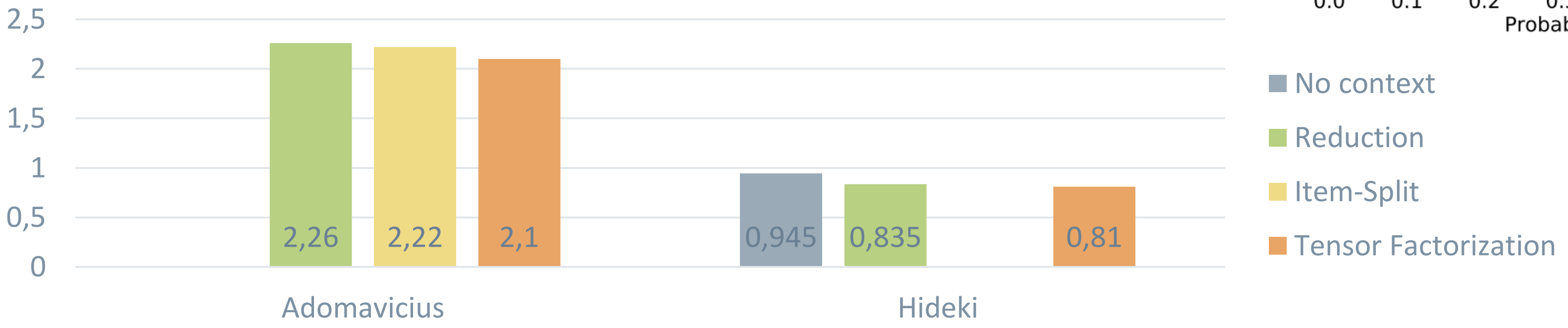


Fig. 4

1
2
3
4

Δt

Comparison of the algorithms

Modelling temporal dynamics	Context-aware Tensor Factorization
Concrete model of time effects	Generalized „framework“ for including context
Can be used for neighbourhood or latent factor methods of Collaborative Filtering	Extension of Matrix Factorization (latent factor method)
Extends base predictors with time dependent components	Approximated factorization (ultimately optimization problem) in N dimensions

Winning method: Collaborative Filtering with Temporal Dynamics

Δt

Thank you for your attention!

Δt

References

Yehuda Koren, Collaborative Filtering with Temporal Dynamics, Communications of the ACM (April 2010), Vol. 53 No. 4

Yehuda Koren, The Bellkor Solution to the Netflix Grand Prize, Netflix prize documentation, 2009

Lei Guo, Matrix Factorization Techniques for Recommender Systems, IRLAB@SDU 2012

Simon Funk, Try this at home, November 2006

Daniel Billsus & Michaels J. Pazzani, Learning Collaborative Information Filters, AAAI 1998

Leskovec, Rajaraman & Ullman, Collaborative Filtering, Stanford University videos

Andrew Ng, Recommender systems: Problem Formulation. Online Open Course

Alexandros Karatzoglou , Xavier Amatriain , Linas Baltrunas , Nuria Oliver, Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering, Proceedings of the fourth ACM conference on Recommender systems, September 26-30, 2010, Barcelona, Spain

G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. ACM Transactions on Information Systems, 23(1):103–145, 2005

C. Ono, Y. Takishima, Y. Motomura, and H. Asoh. Context-aware preference model based on a study of difference between real and supposed situation data. In UMAP ’09: Proceedings of the 17th International Conference on User Modeling, Adaptation, and Personalization, pages 102–113, Berlin, Heidelberg, 2009. Springer-Verlag