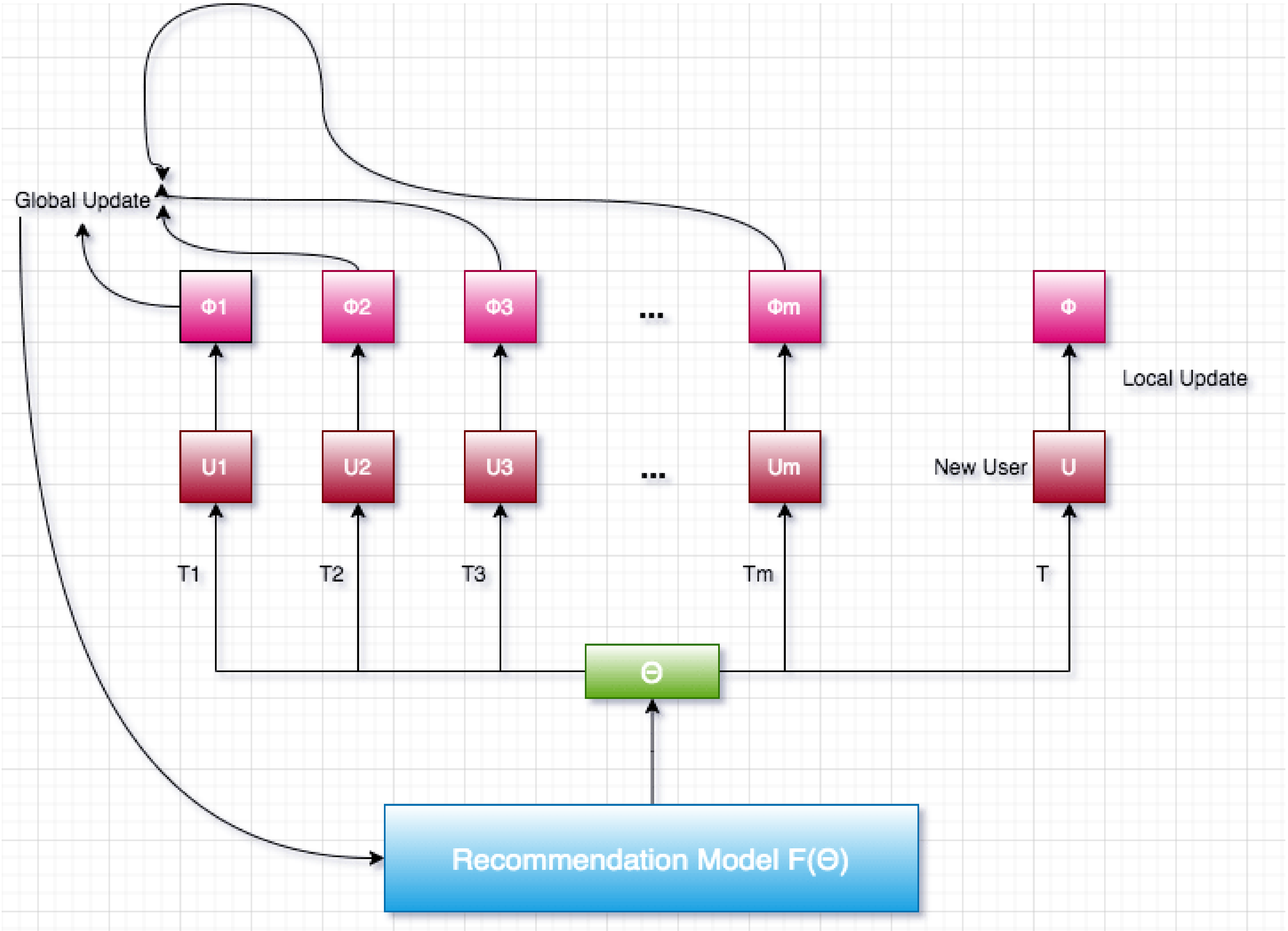
Master's Thesis Defense

MetaRec
Meta-Learning Meets Recommendation Systems

Presented by **James Le**
Advised by **Alexander Ororbia II**
As Member of **Neural Adaptive Computing Lab**

ROCHESTER INSTITUTE OF TECHNOLOGY
R·I·T
1829

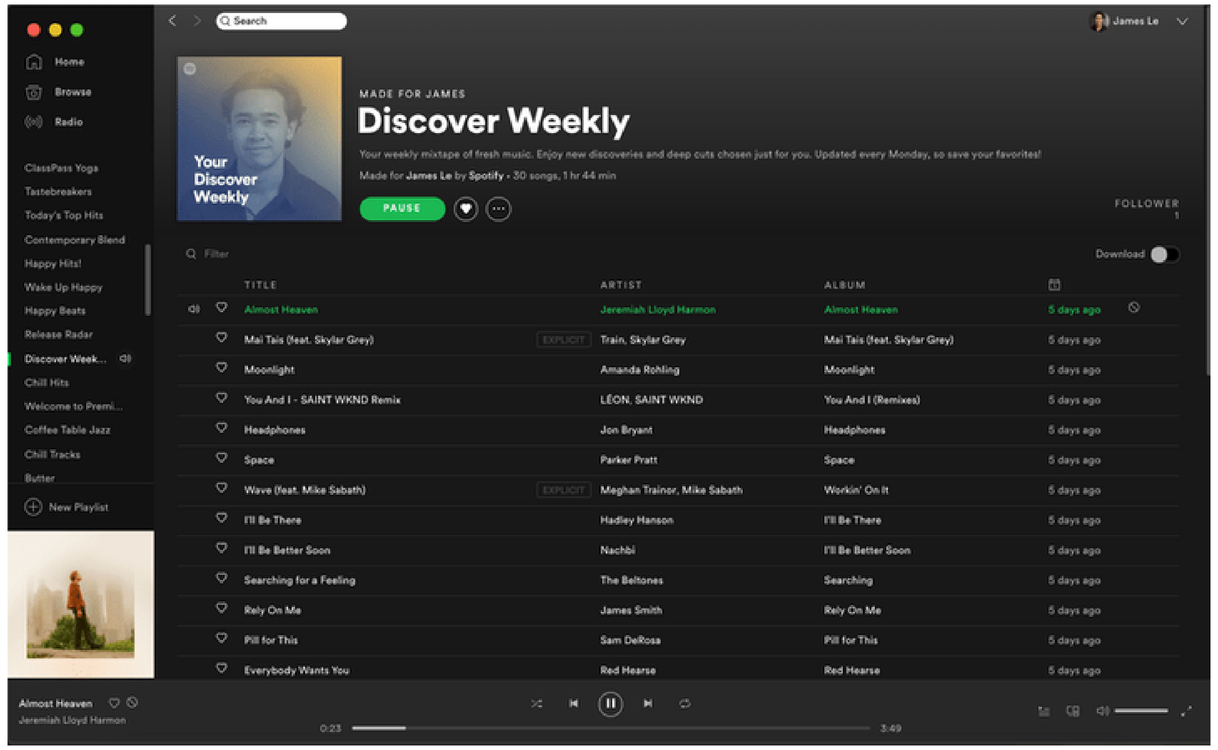
Outline



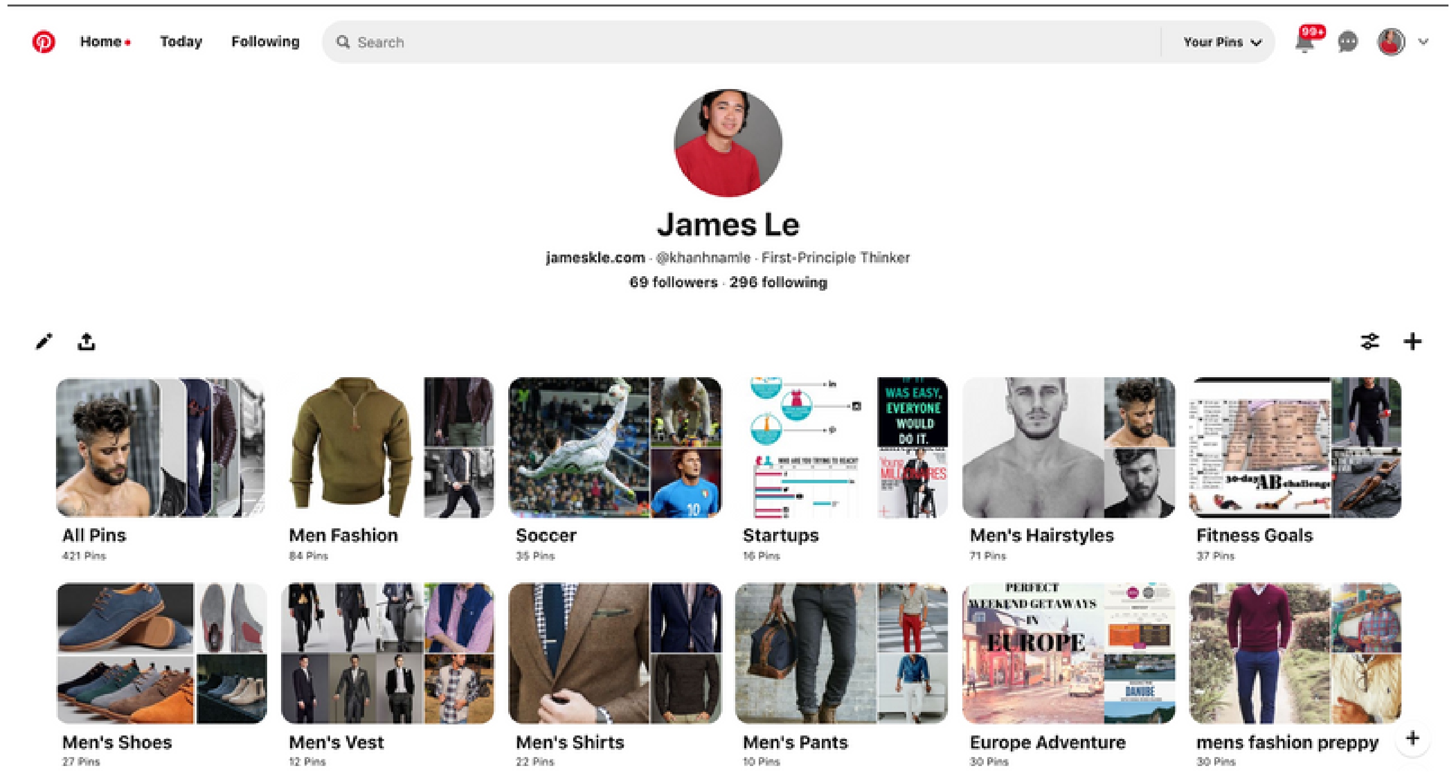
Global Update
Φ1
Φ2
Φ3
...
Φm
Φ
Local Update
U1
U2
U3
...
Um
New User
U
T1
T2
T3
Tm
T
Θ
Recommendation Model F(Θ)

Introduction

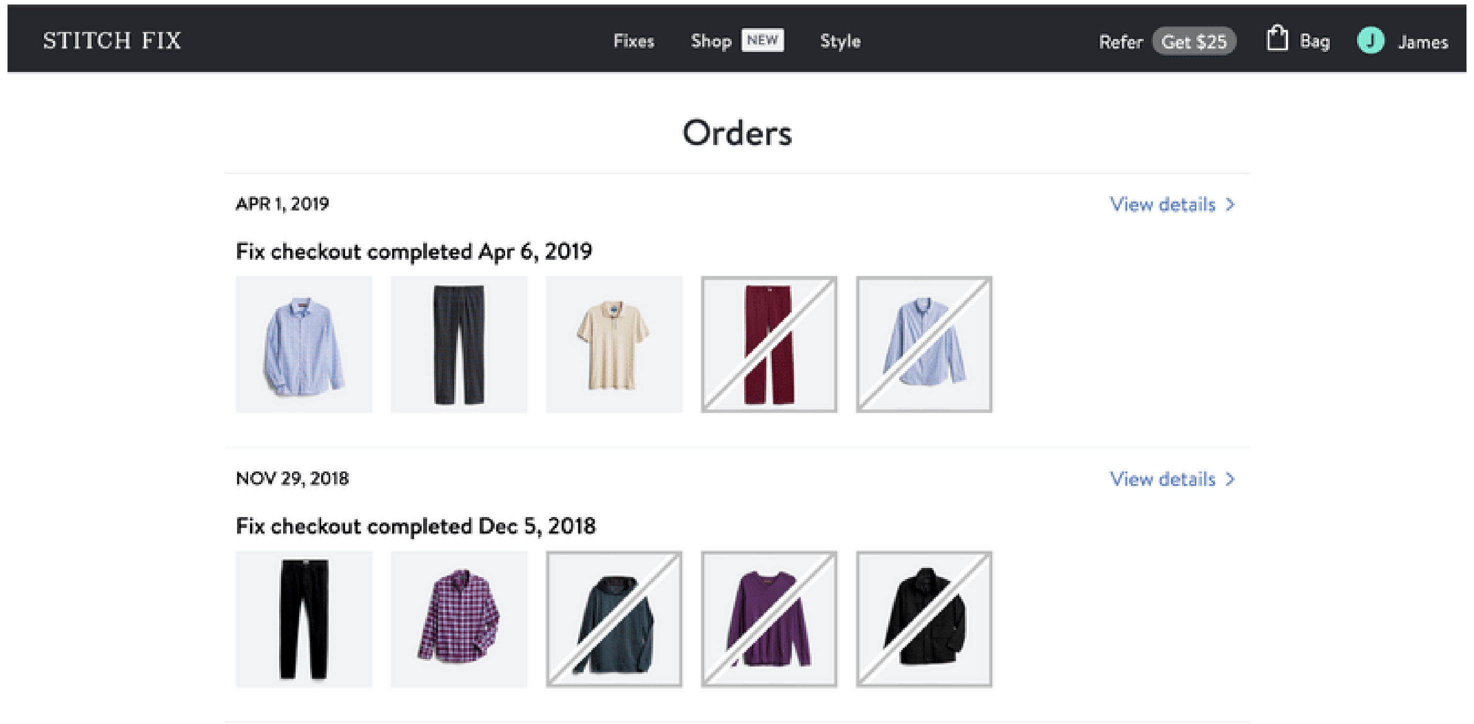
PART 1



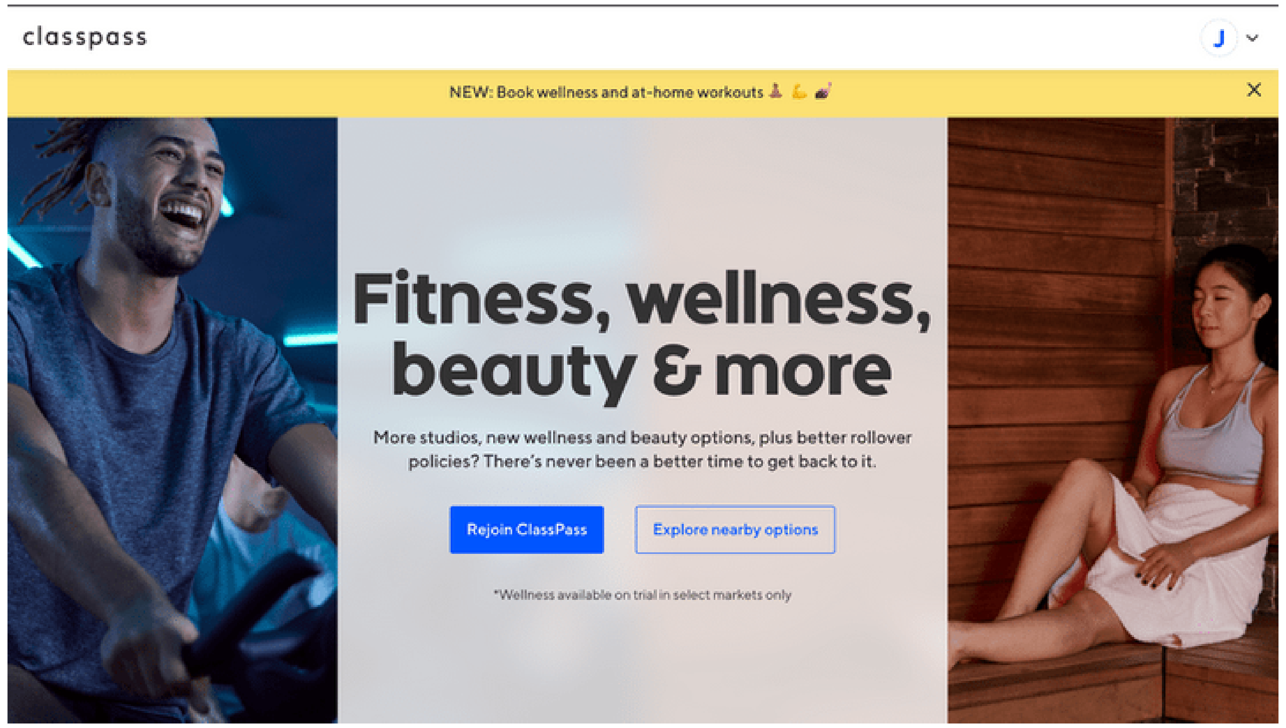
ENTERTAINMENT



CONTENT



E-COMMERCE



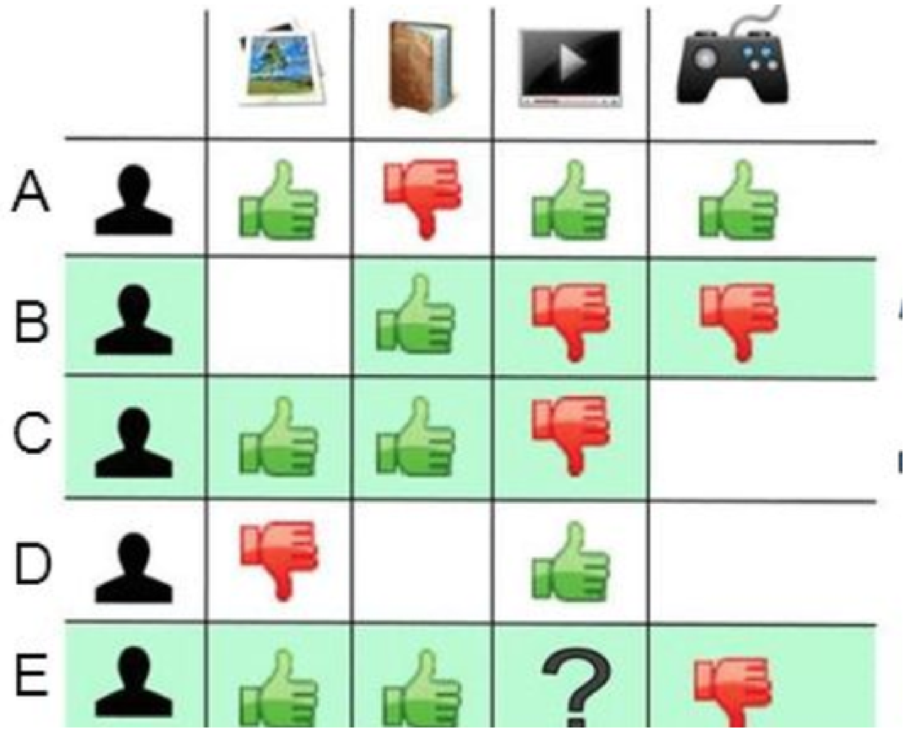
SERVICES

Challenges

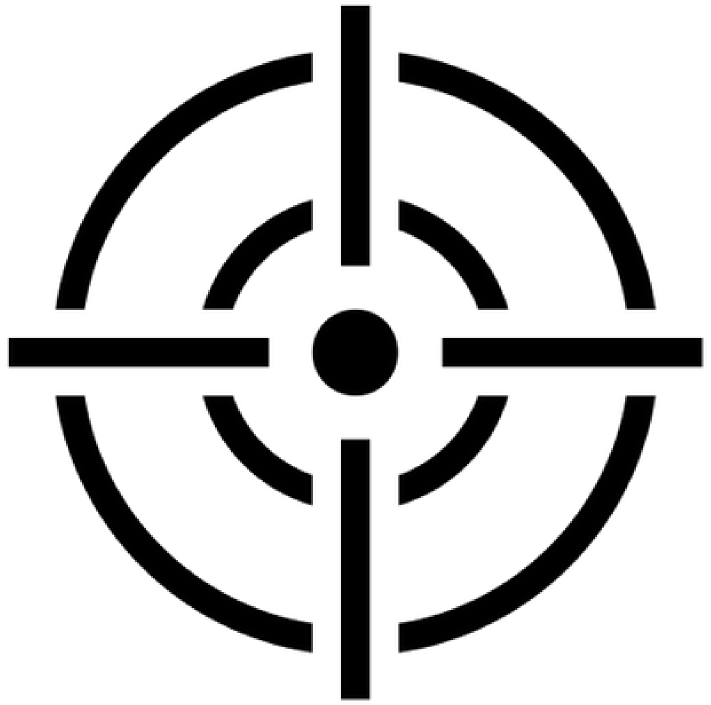
WHAT I WANT TO TACKLE

Scalability

Sparsity



Accuracy



Challenges

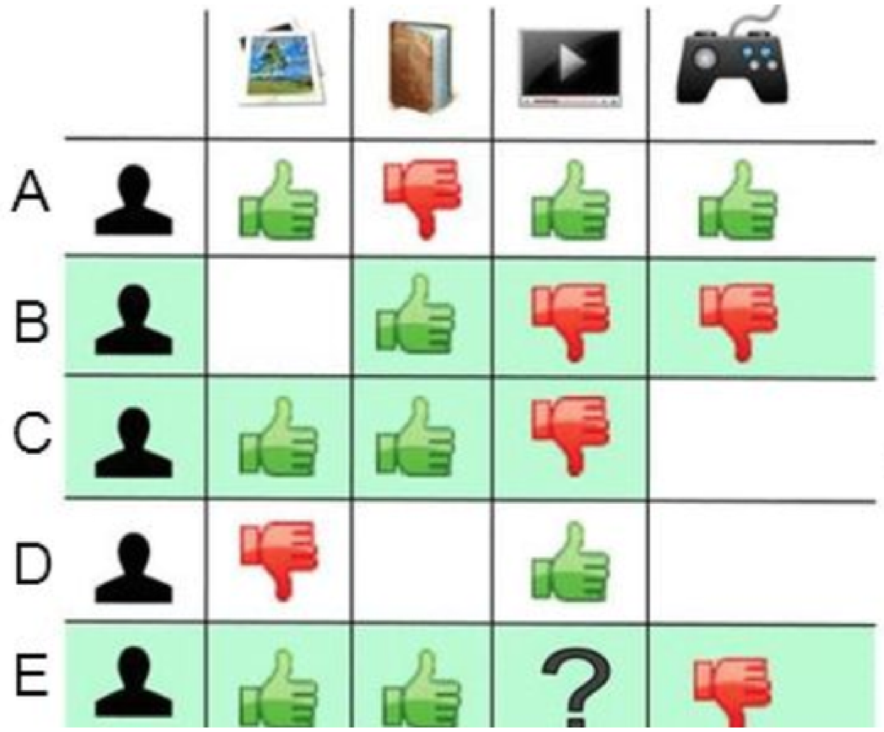
WHAT I WANT TO TACKLE

→

Scalability

→

Sparsity



→

Accuracy

"RECOMMENDATION SYSTEMS THAT META-LEARN INFORMATION FROM RELATED TASKS OUTPERFORM SYSTEMS THAT DO NOT USE THIS INFORMATION ACROSS A WIDE RANGE OF TASKS IN ACCURACY METRICS."

Recommendation Systems

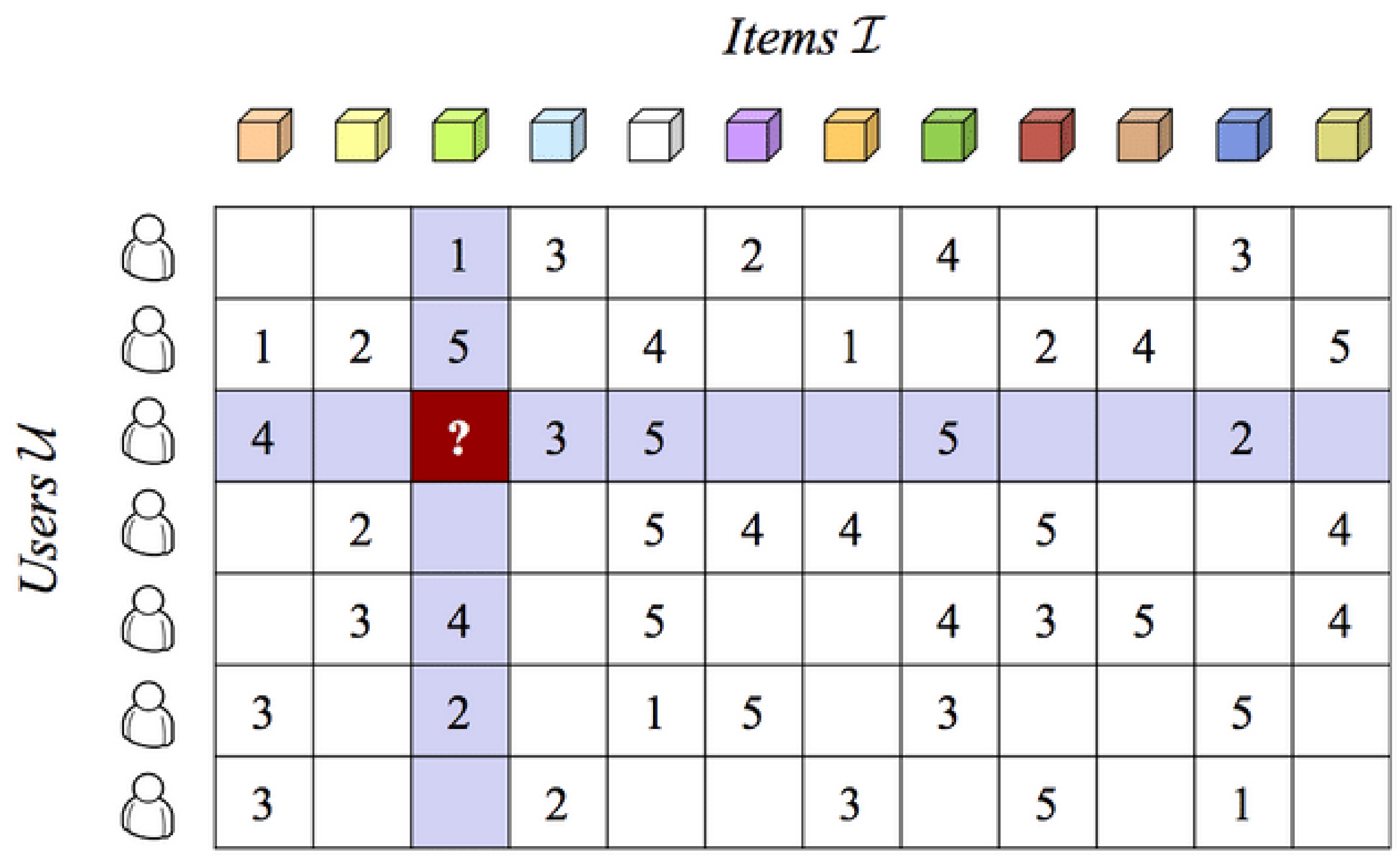
PART 2

BY JAMES LE

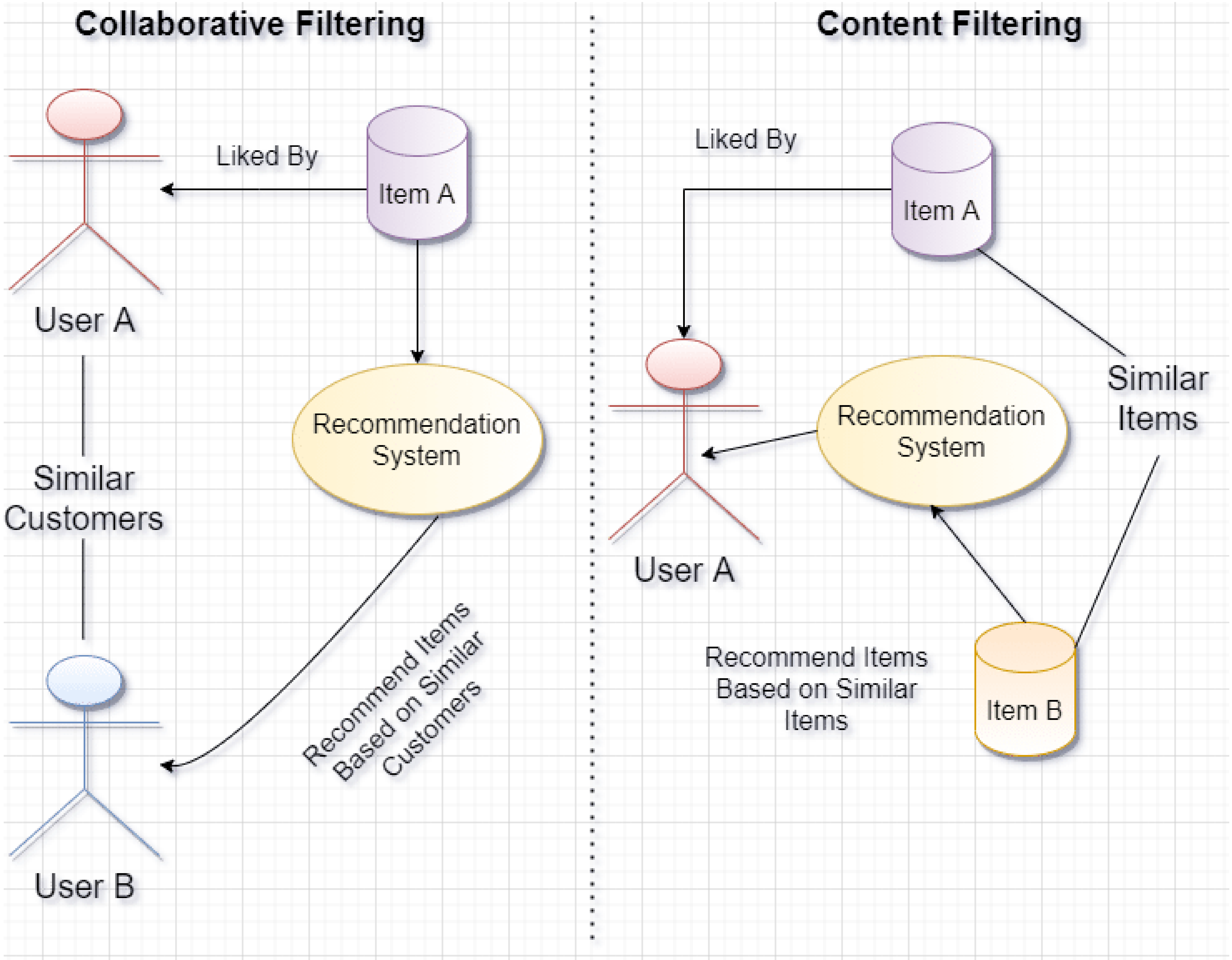
Problem Formulation

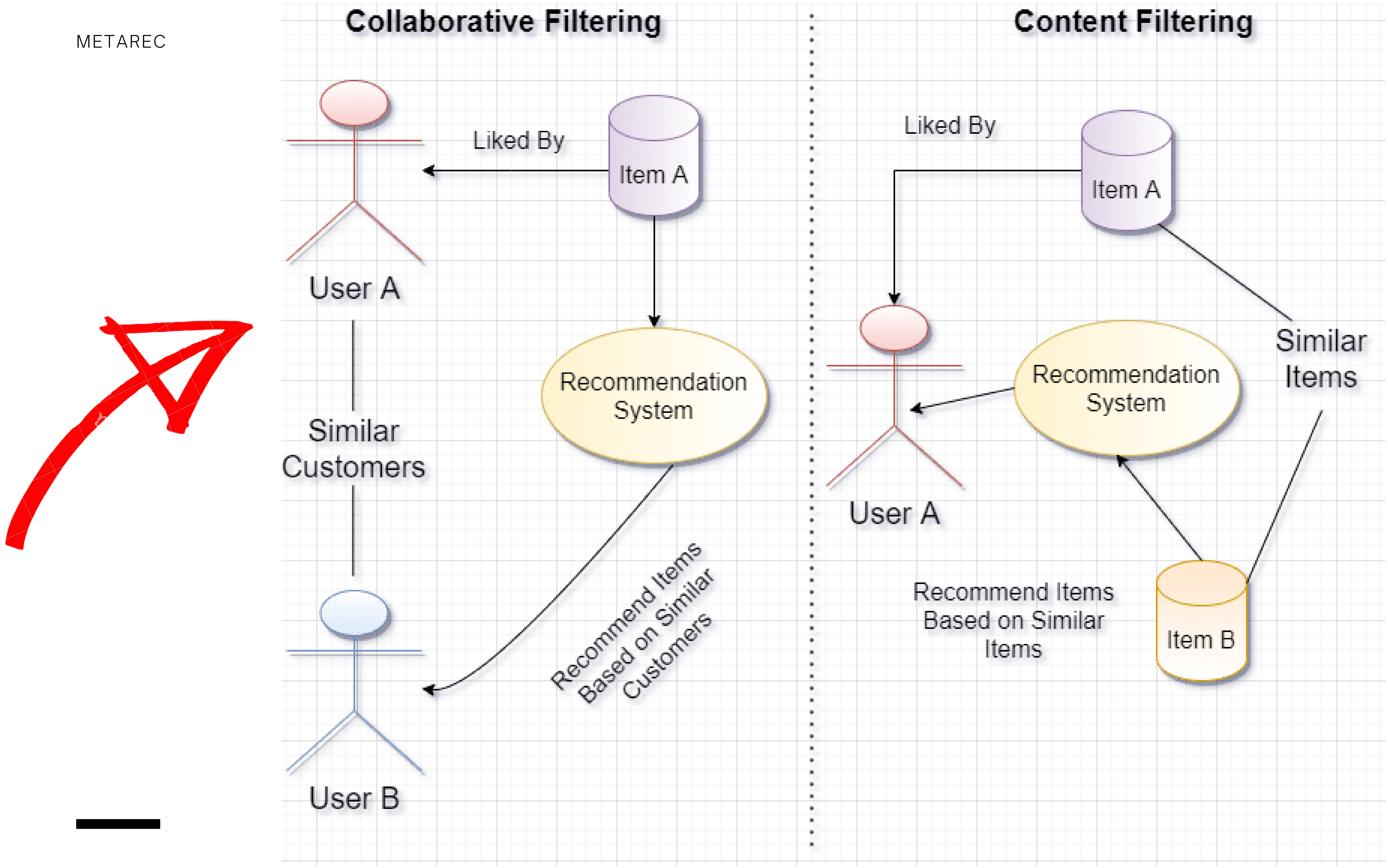
$$\forall u \in U, i_s = \arg\max_{i \in I} F(u, i)$$

- **U** - set of all users
- **I** - set of all items
- **F(u, i)** - utility function that measures the relevance of item i to user u
- **F** is usually represented as a rating in numeric scale

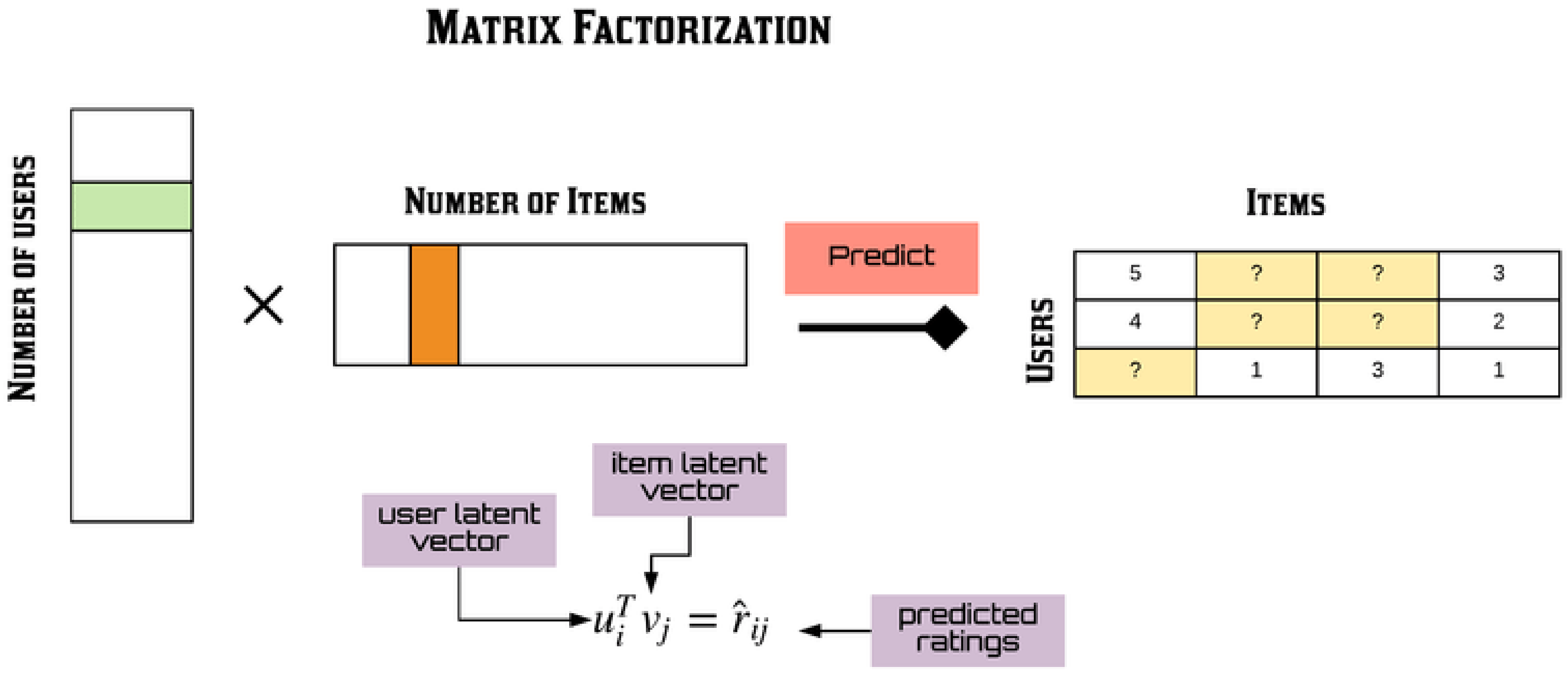


		1	3		2		4			3	
1	2	5		4		1		2	4		5
4		?	3	5			5			2	
	2			5	4	4		5			4
	3	4		5			4	3	5		4
3		2		1	5		3			5	
3			2			3		5		1	


Collaborative Filtering
Liked By
Item A
User A
Recommendation System
Similar Customers
Recommend Items Based on Similar Customers
User B
Content Filtering
Liked By
Item A
Similar Items
Recommendation System
User A
Recommend Items Based on Similar Items
Item B

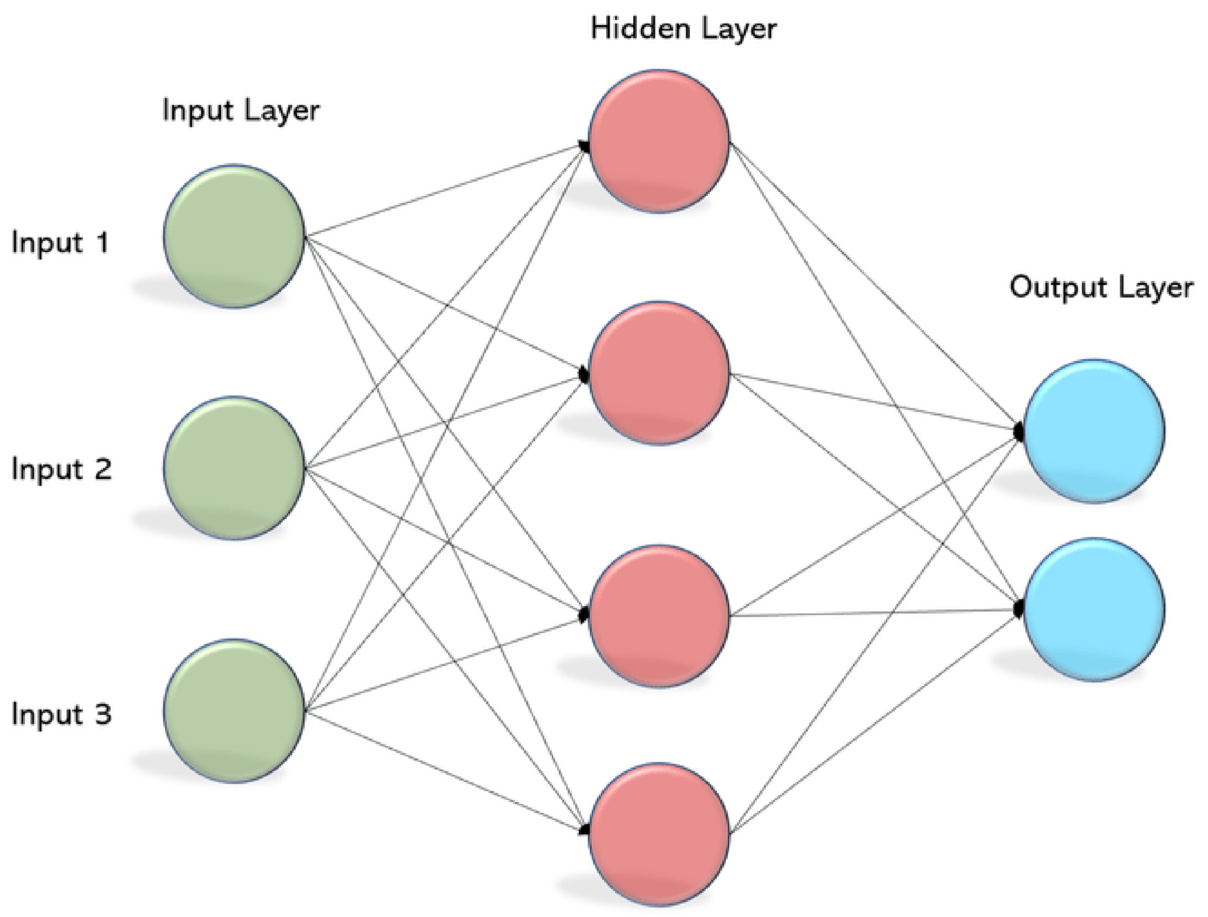

Collaborative Filtering
Liked By
Item A
User A
Recommendation System
Similar Customers
Recommend Items Based on Similar Customers
User B
Content Filtering
Liked By
Item A
User A
Recommendation System
Similar Items
Recommend Items Based on Similar Items
Item B

Matrix Factorization

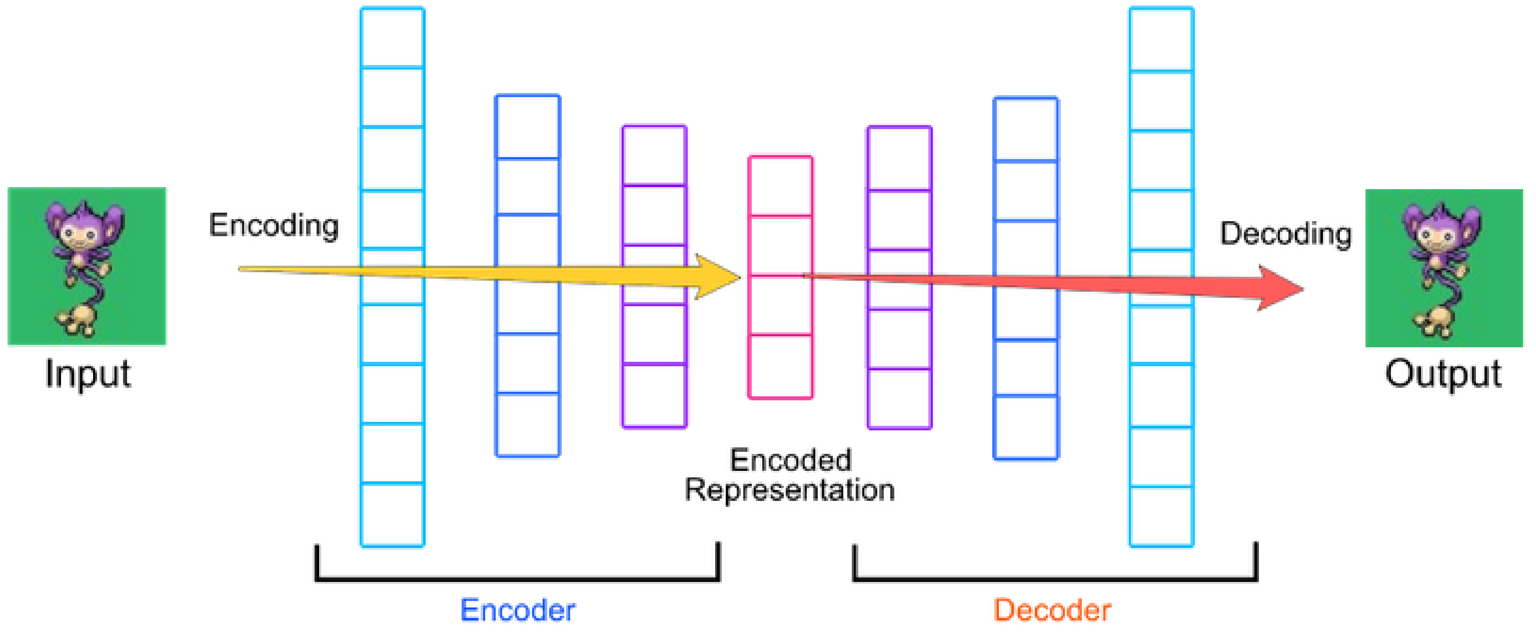


- The de-facto standard for collaborative filtering
- Represent user preferences as an user-item matrix
- Find two smaller matrices (latent embeddings) that approximate the full matrix
- Minimize the rating prediction errors
- Can be optimized via Gradient Descent
- Prediction is simple

Neural Networks



MULTI-LAYER PERCEPTRON



AUTOENCODER

Neural Networks For Recommendations

SOLUTIONS FOR EXISTING CHALLENGES

SCALABILITY → **SPARSITY** → **ACCURACY**

SCALABILITY

- Extract low-dimensional factors of high-dimensional user preferences for the items

SPARSITY

- Engineer content-based features and integrate them into the algorithm
- Extract high-level representations of user preferences for the items

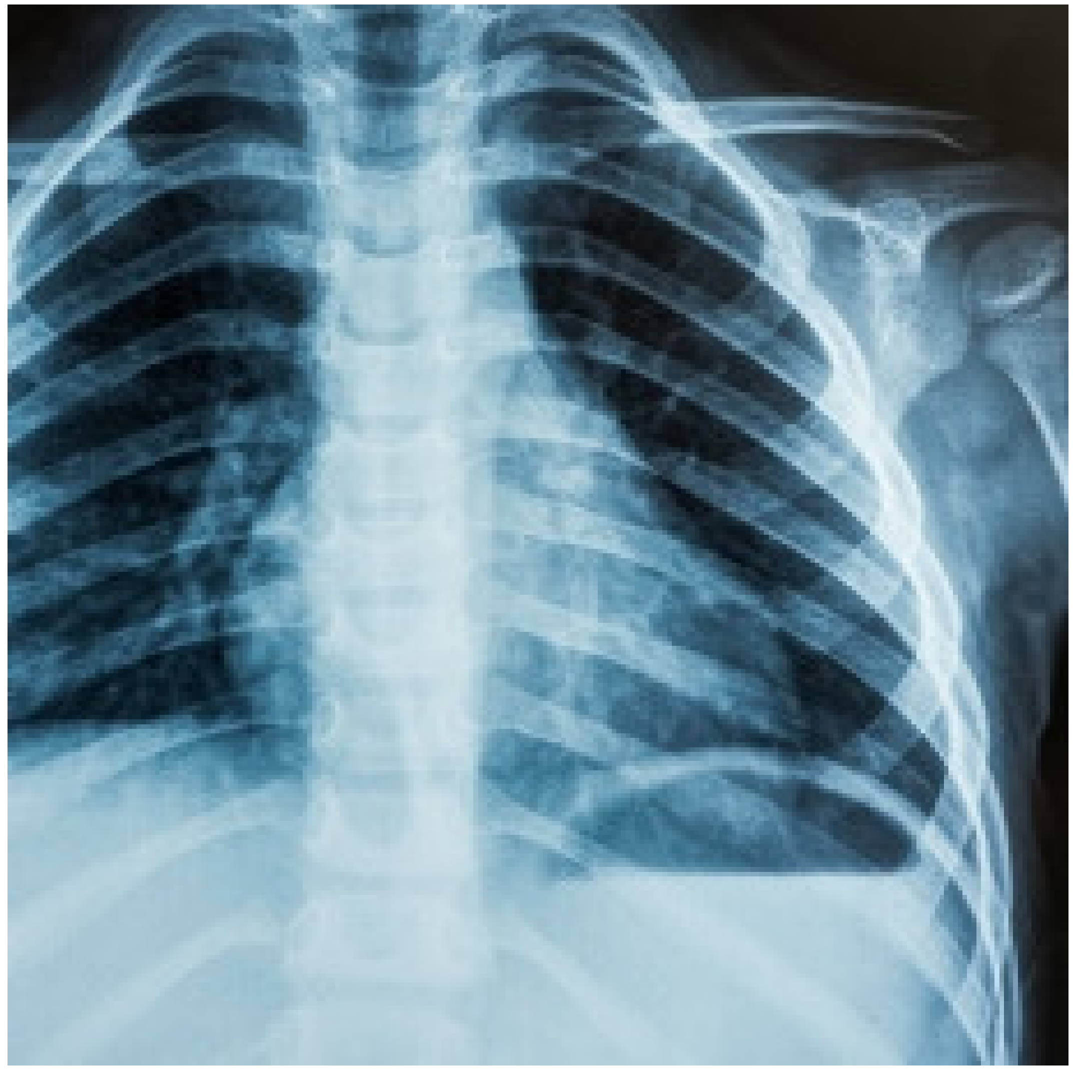
ACCURACY

- Extract user and item's latent factors
- Jointly combine information from different data sources and modalities

Meta Learning

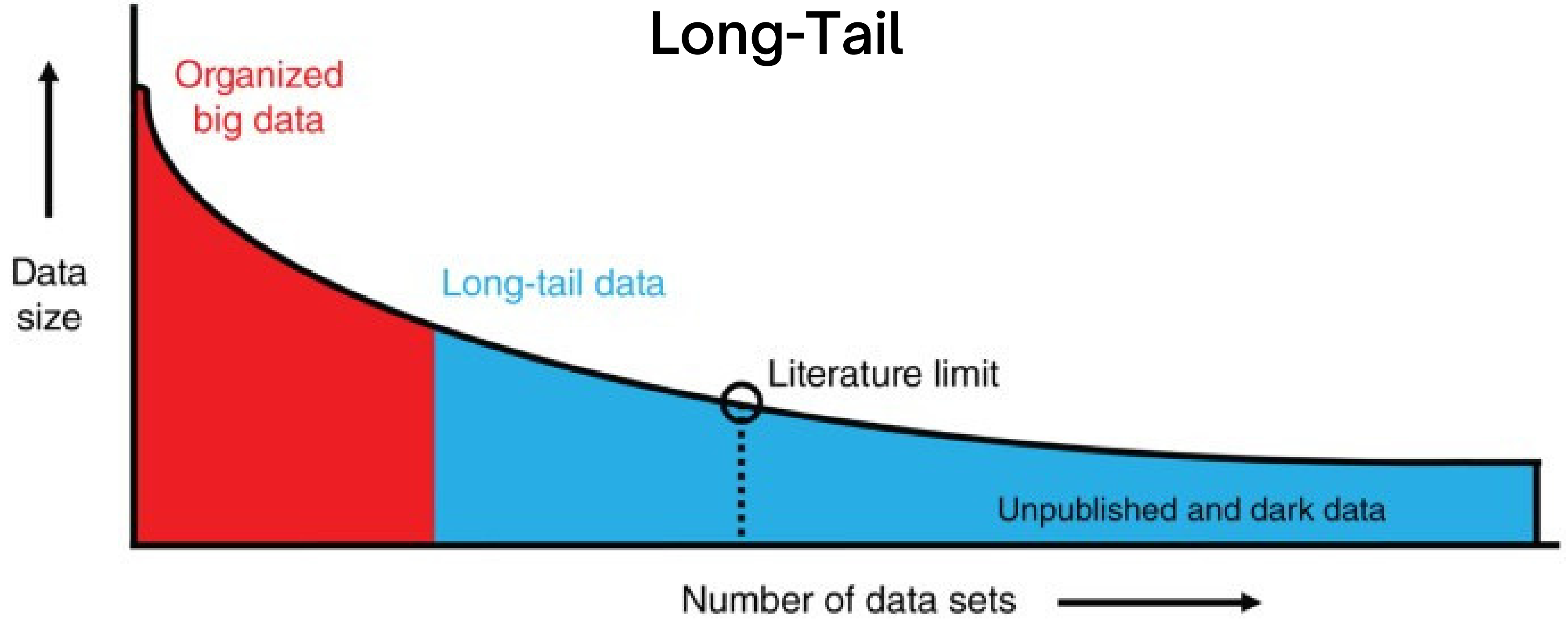
PART 3

Motivation



Limited Data

Long-Tail



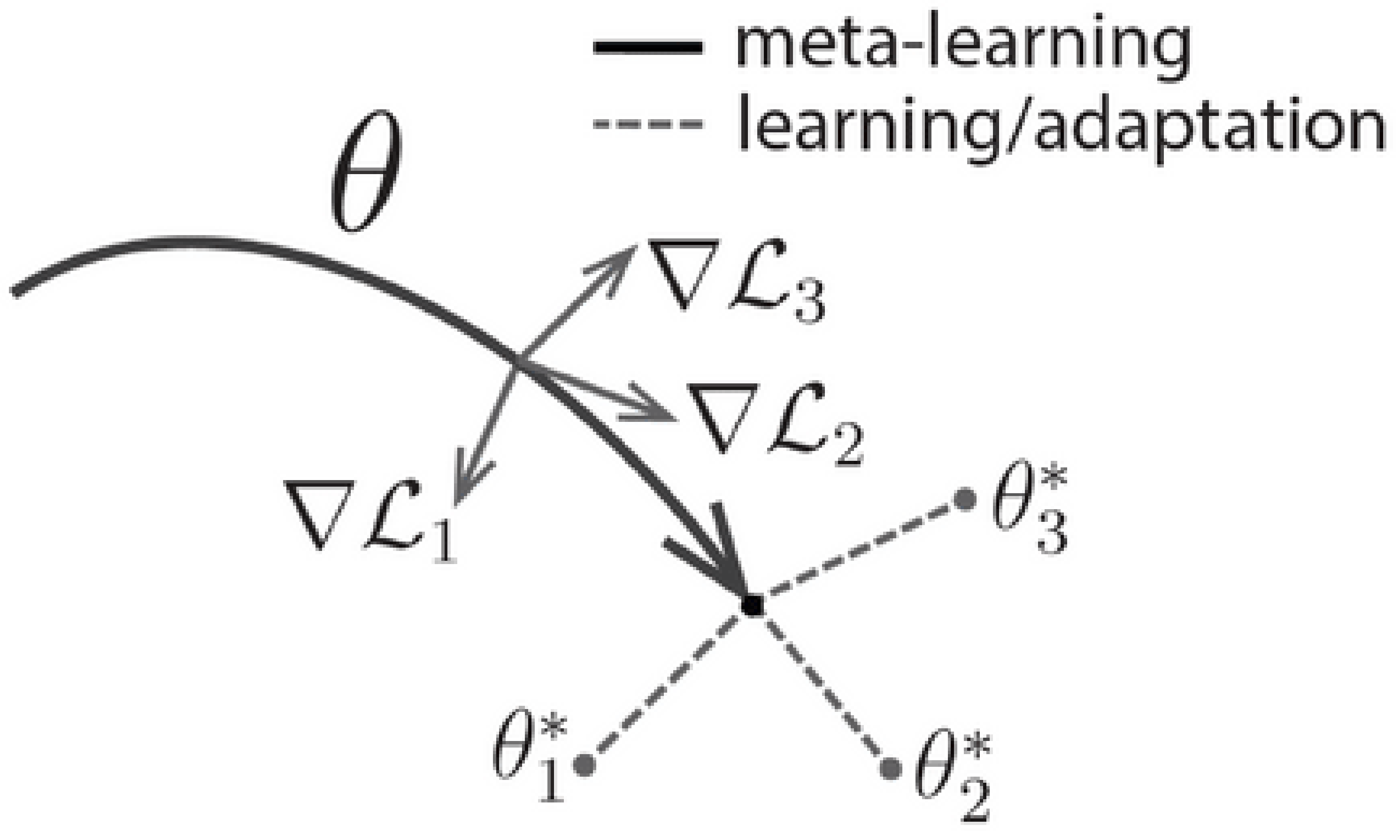
Fast Inference

Model-Agnostic Meta-Learning (MAML, Finn et. al, 2017)

$$\min_{\theta} \sum_{\text{tasks } i} \mathcal{L}^{(i)}_{\text{val}}(\theta - \alpha \nabla_{\theta} \mathcal{L}^{(i)}_{\text{train}}(\theta))$$

θ parameter vector being meta-learned

θ^*_i optimal parameter vector for task i



MetaRec

PART 4

$$\text{Binary Classification: } P(r_{ui} = 1 | p_u, q_i) = F_\theta(p_u, q_i, \theta)$$

$$\text{Rating Regression: } \hat{r}_{ui} = F_\theta(p_u, q_i, \theta)$$

$$\phi_u = \theta - \alpha \nabla_\theta L_{T_u}(F_\theta)$$

$$\theta = \theta - \beta \nabla_\theta \sum_{T_u \sim B} L_{T_u}(F_{\phi_u})$$

RECOMMENDATION TASK

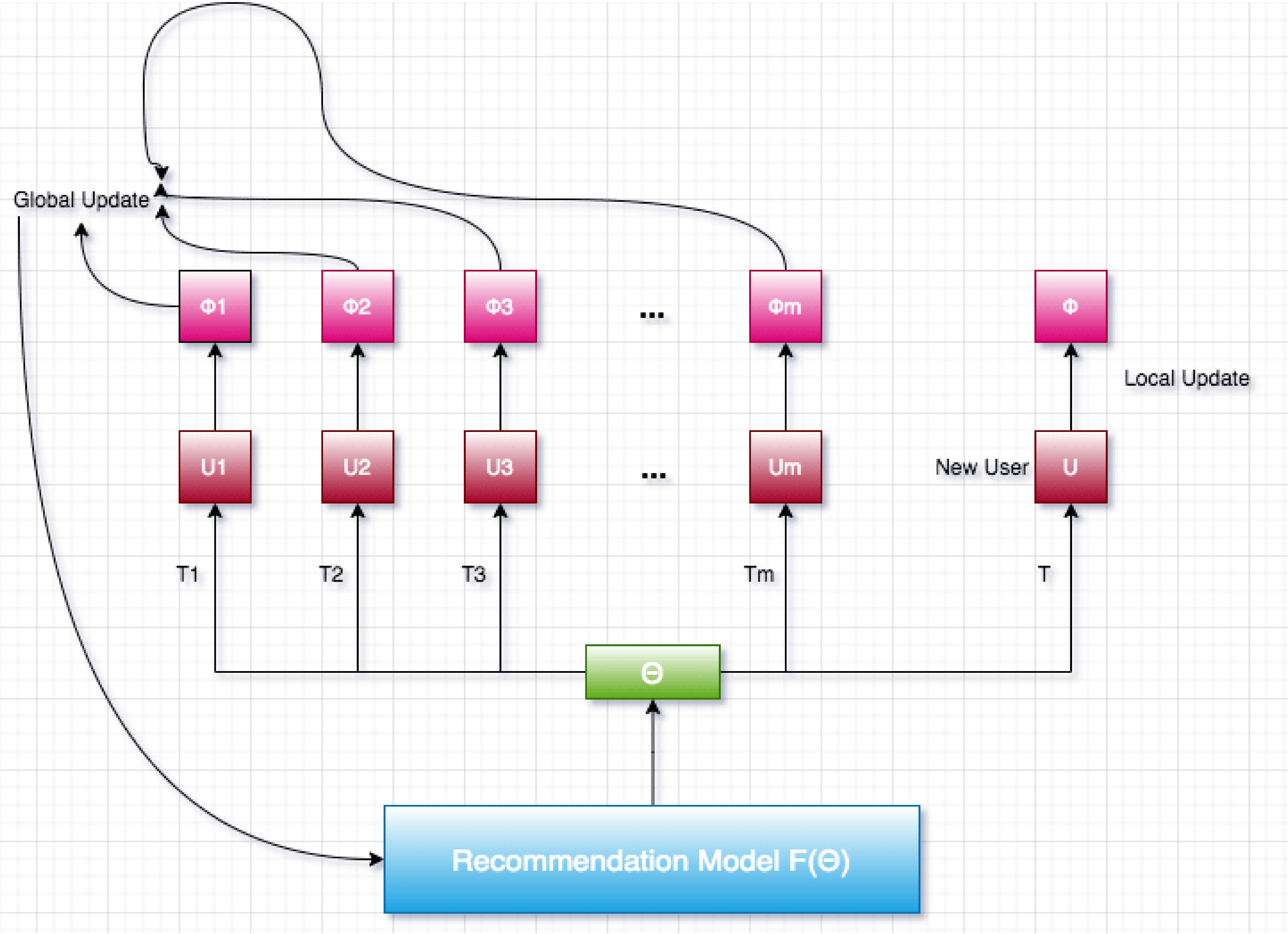
- Task T = Recommend items to one user given examples of items already rated by that user
- Binary Classification (Implicit Feedback)
- Rating Regression (Explicit Feedback)
- θ denotes model parameters

LOCAL UPDATE

- Sample a batch of tasks B
- Uses MAML to compute task-specific parameters φ based on the loss function value for each task T in batch B
- α is the local learning rate

GLOBAL UPDATE

- Uses MAML to compute model parameters θ on the sum of test losses for all tasks T
- β is the global learning rate


Global Update
Φ1
Φ2
Φ3
...
Φm
Φ
Local Update
U1
U2
U3
...
Um
New User
U
T1
T2
T3
Tm
T
Θ
Recommendation Model F(Θ)

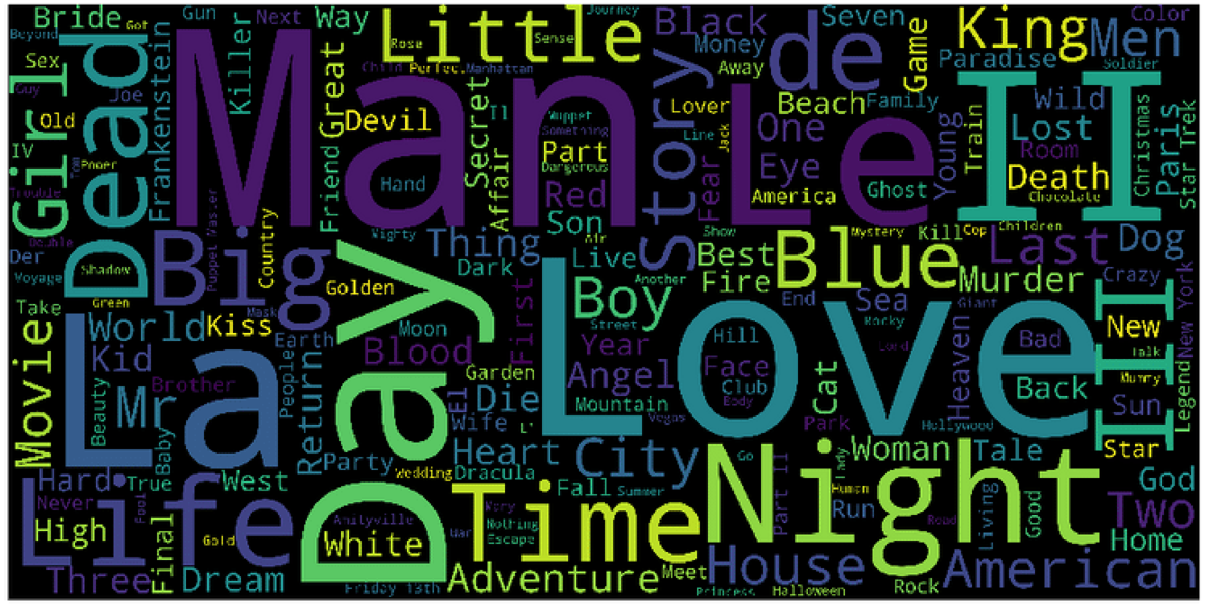
Experiments

PART 5

MovieLens1M

(HARPER AND KONSTAN, 2015)

	title	genres	rating
0	Toy Story (1995)	Animation|Children's|Comedy	5
489283	American Beauty (1999)	Comedy|Drama	5
489259	Election (1999)	Comedy	5
489257	Matrix, The (1999)	Action|Sci-Fi|Thriller	5
489256	Dead Ringers (1988)	Drama|Thriller	5
489237	Rushmore (1998)	Comedy	5
489236	Simple Plan, A (1998)	Crime|Thriller	5
489226	Hands on a Hard Body (1996)	Documentary	5
489224	Pleasantville (1998)	Comedy	5
489212	Say Anything... (1989)	Comedy|Drama|Romance	5
489207	Beetlejuice (1988)	Comedy|Fantasy	5
489190	Roger & Me (1989)	Comedy|Documentary	5
489172	Buffalo 66 (1998)	Action|Comedy|Drama	5
489171	Out of Sight (1998)	Action|Crime|Romance	5
489170	I Went Down (1997)	Action|Comedy|Crime	5
489168	Opposite of Sex, The (1998)	Comedy|Drama	5
489157	Good Will Hunting (1997)	Drama	5
489152	Fast, Cheap & Out of Control (1997)	Documentary	5
489149	L.A. Confidential (1997)	Crime|Film-Noir|Mystery|Thriller	5
489145	Contact (1997)	Drama|Sci-Fi	5



Attributes	Users	Movies	Ratings
Total	6,040	3,883	1,000,209
Min Rating	20	1	1
Max Rating	2,314	3,428	5
Average Rating	165.6	269.9	3.58

Matrix Factorization Experiments

BASELINE MODELS

Vanilla Matrix Factorization
(Koren et al., 2009)

$$r_{ui} = p_u \cdot q_i^T$$

Matrix Factorization with Biases
(Koren et al., 2009)

$$r_{ui} = b + \omega_u + \omega_i + p_u \cdot q_i^T$$

Matrix Factorization with Side Features
(Koren et al., 2009)

$$r_{ui} = b + d_o + \omega_u + \omega_i + (p_u + t_o) \cdot q_i^T$$

Matrix Factorization with Temporal Features
(Koren et al., 2009)

$$r_{ui} = b + \omega_u + \omega_i + p_u \cdot q_i^T + m_u \cdot n_t$$

Matrix Factorization Experiments

BASELINE MODELS

Factorization Machine
(Rendle, 2010)

$$r_{ui} = b + \sum_{i=1}^{n} \omega_i \cdot x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} < v_i, v_j > x_i \cdot x_j \qquad \mathbf{w} \in \mathbb{R}^n, \quad \mathbf{V} \in \mathbb{R}^{n \times k}$$

Matrix Factorization with Mixture of Taste
(Kula, 2018)

$$r_{ui} = \sigma(A_u \cdot q_i^T) \cdot (U_u \cdot q_i^T) + b + \omega_u + \omega_i$$

$$\sigma(x_i) = \frac{\exp^{x_i}}{\sum_j \exp^{x_j}} \qquad A \in \mathbb{R}^{m \times k}, U \in \mathbb{R}^{m \times k}$$

Variational Matrix Factorization
(Porteus et al., 2010, Kim et al., 2014)

$$r_{ui} = b + \omega_u + \omega_i + GS(\mu_u, v_u) \cdot GS(\mu_i, v_i)$$

$$GS(\mu, v) = \mu + \mathcal{N}(0,\, 1) \cdot \sqrt{v}$$

MetaRec-MF

BASE MODEL

Matrix Factorization with Biases

$$\hat{r}_{ui} = F(u, i | p_u, q_i, \theta) = p_u \cdot q_i + b + w_u + w_i$$

LOSS FUNCTION

Mean-Squared Error with L2 Regularization

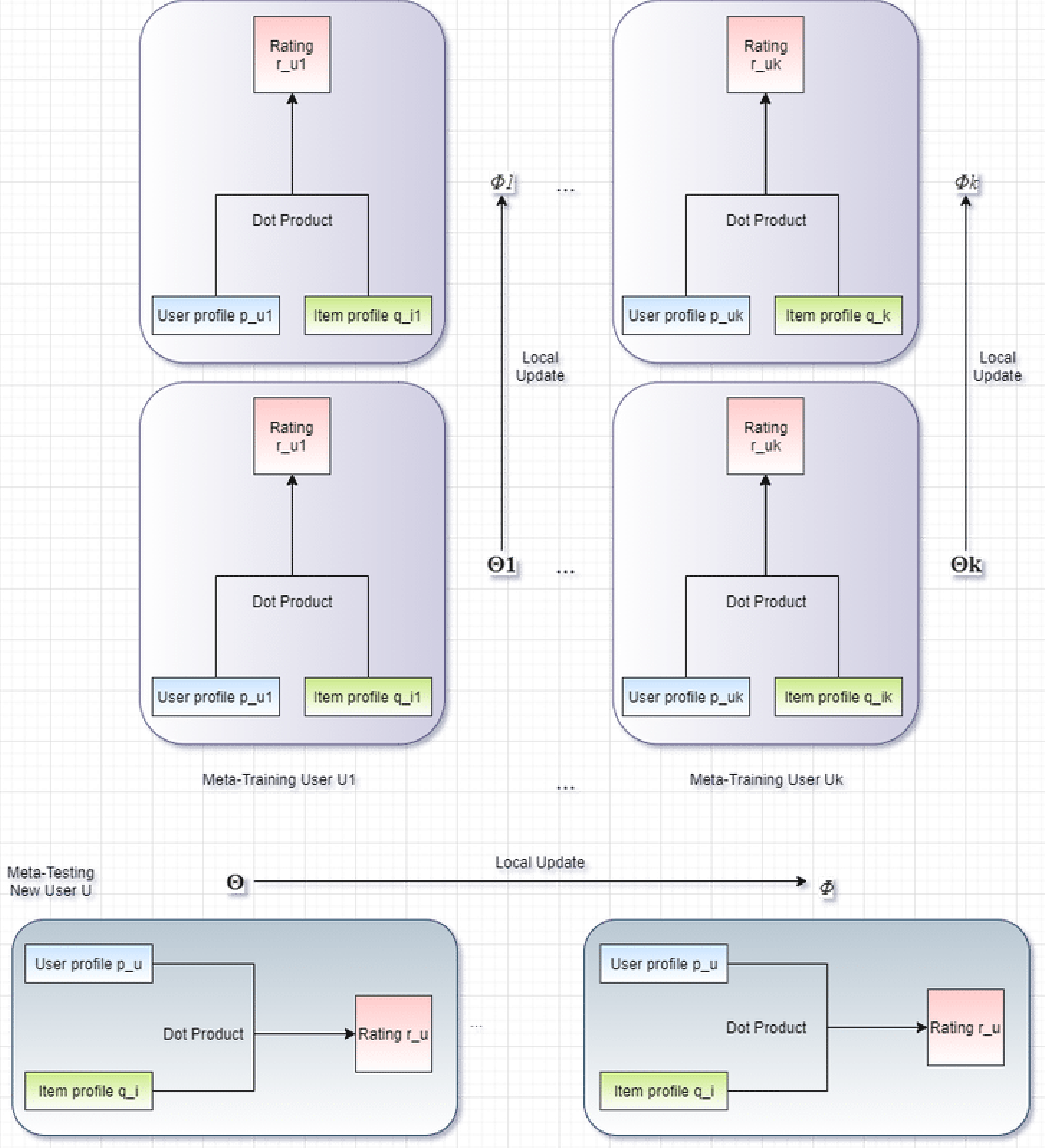
$$L(\theta) = \frac{1}{|D_{train}|} \sum_{r_{u,i} \in D_{train}} (r_{u,i} - \hat{r}_{u,i})^2 + \lambda \frac{1}{2} ||\theta||_2^2$$

EVALUATION METRICS

- Rating Regression with Explicit Feedback
- Mean Squared Error
- Mean Absolute Error

$$MAE = \frac{1}{|D_{test}|} \sum_{r_{u,i} \in D_{test}} |r_{u,i} - \hat{r}_{u,i}|$$

$$MSE = \frac{1}{2|D_{test}|} \sum_{r_{u,i} \in D_{test}} (r_{u,i} - \hat{r}_{u,i})^2$$


Rating r_u1
Dot Product
User profile p_u1
Item profile q_i1
Rating r_u1
Dot Product
User profile p_u1
Item profile q_i1
Φi
Local Update
Θ1
...
Rating r_uk
Dot Product
User profile p_uk
Item profile q_k
Rating r_uk
Dot Product
User profile p_uk
Item profile q_ik
Φk
Local Update
Θk
Meta-Training User U1
Meta-Training User Uk
Meta-Testing New User U
Θ
Local Update
Φ
User profile p_u
Dot Product
Item profile q_i
Rating r_u
User profile p_u
Dot Product
Item profile q_i
Rating r_u

Regression Results

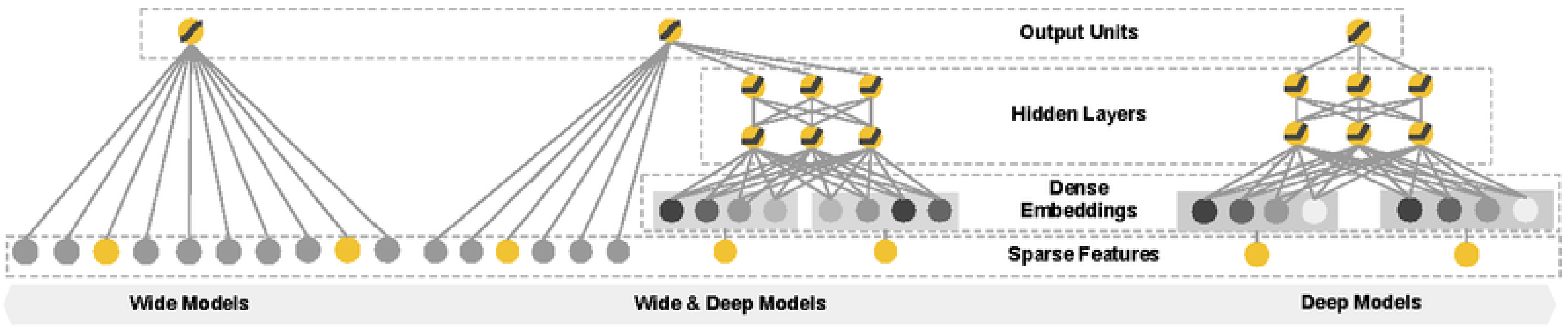
Models	MAE	MSE	Training Time
MF	0.703	0.817	**6m5s**
MF-Bias	0.694	0.790	11m38s
MF-Side	0.691	**0.784**	13m34s
MF-Temporal	0.689	0.793	18m51s
FM	0.712	0.823	**3m40s**
MF-Mixture	**0.687**	0.788	13m44s
Variational-MF	0.707	0.839	16m51s
MetaRec-MF (Ours)	**0.687**	**0.760**	12m45s

- Train-Test Split: 75-25
- Trained for 50 Epochs
- MetaRec-MF outperforms the other models on both MAE and MSE metrics (Lower values are better)
- Slight tradeoff between accuracy performance and compute cost
- The Factorization Machine model took the fastest time to train
- Adding more features to the matrix factorization equation led to longer training time

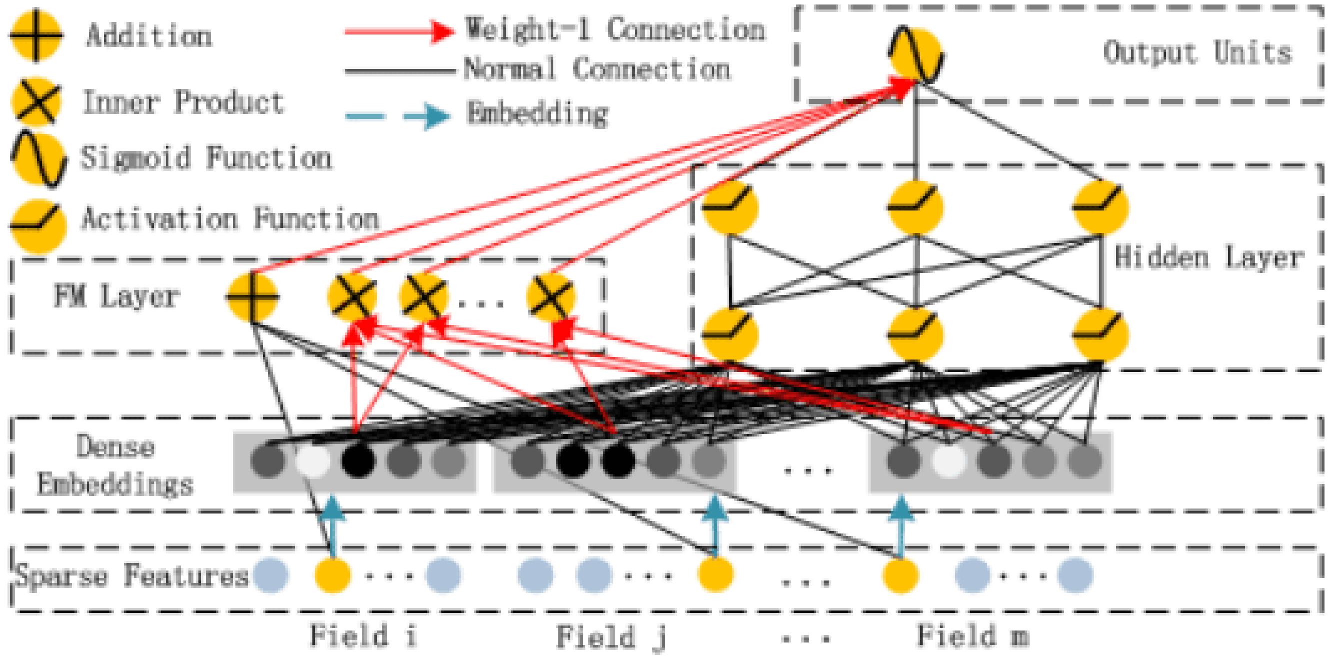
Multi-Layer Perceptron Experiments

BASELINE MODELS

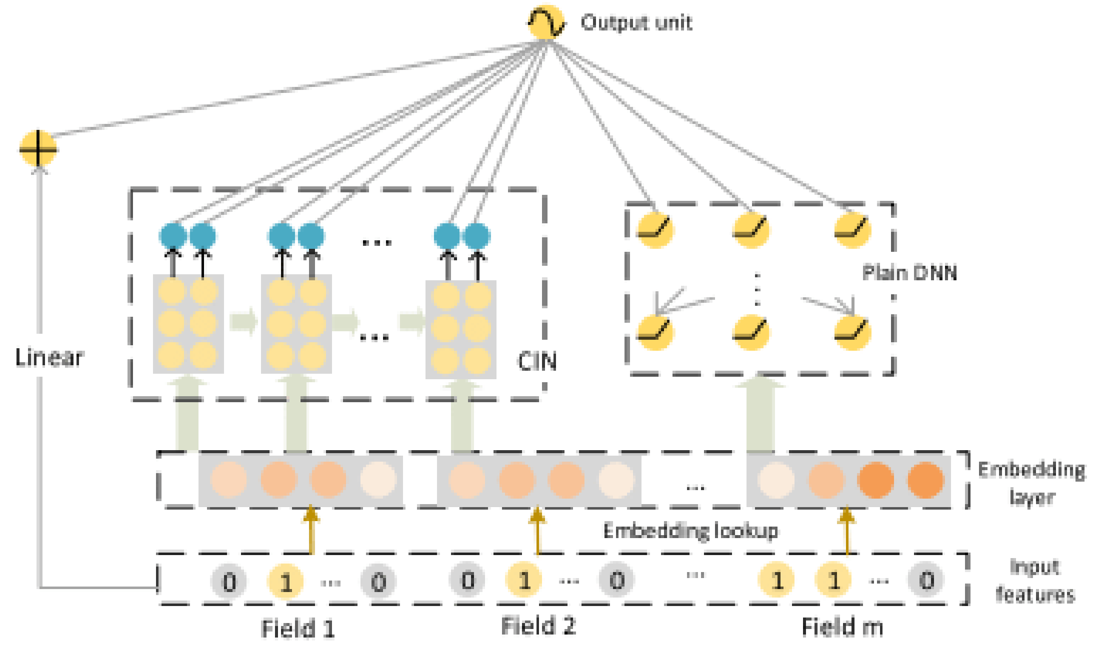
Wide and Deep (Cheng et al., 2016)



Deep Factorization Machine (Guo et al., 2017)



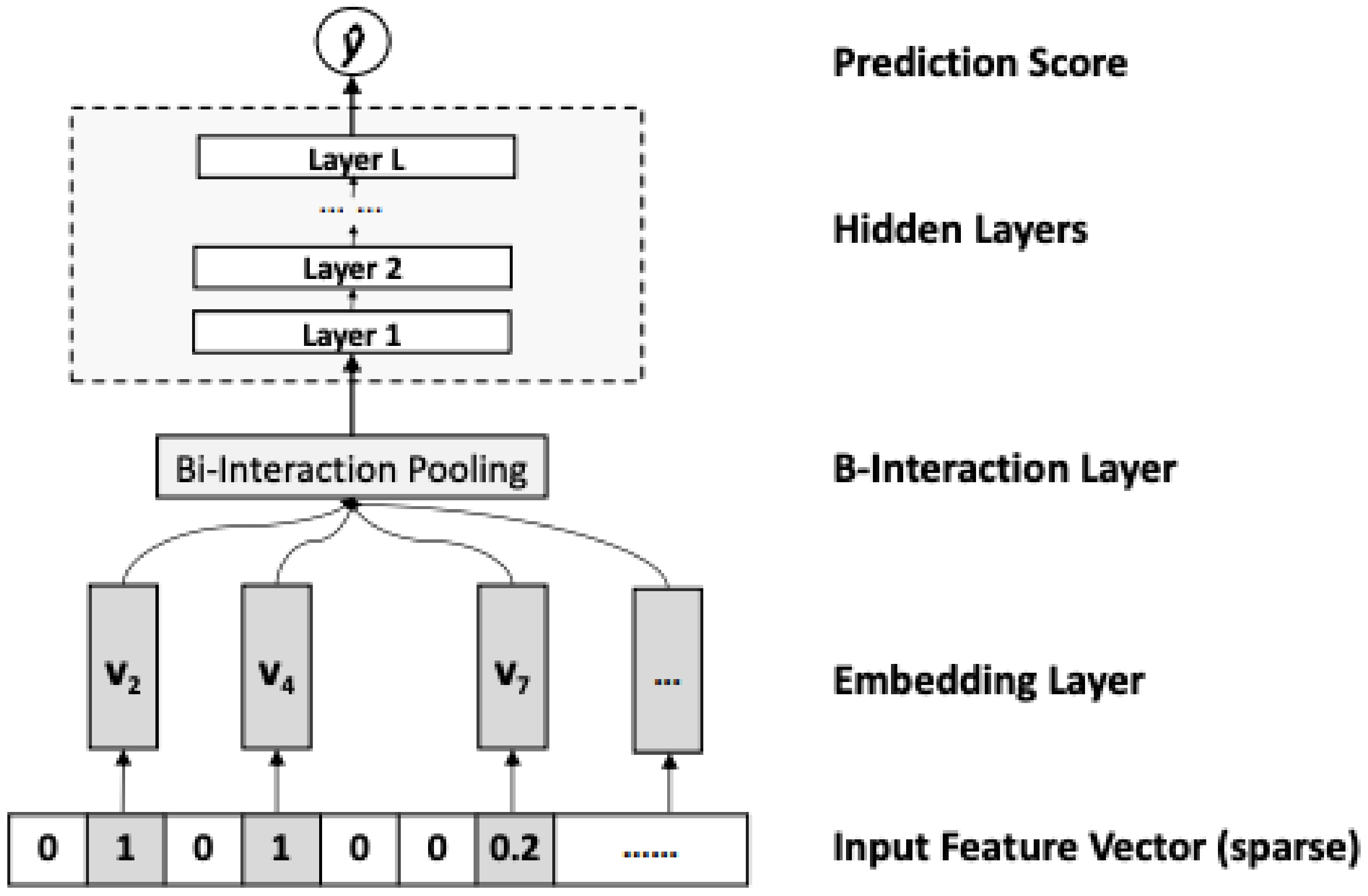
Extreme Deep Factorization Machine (Lian et al., 2018)



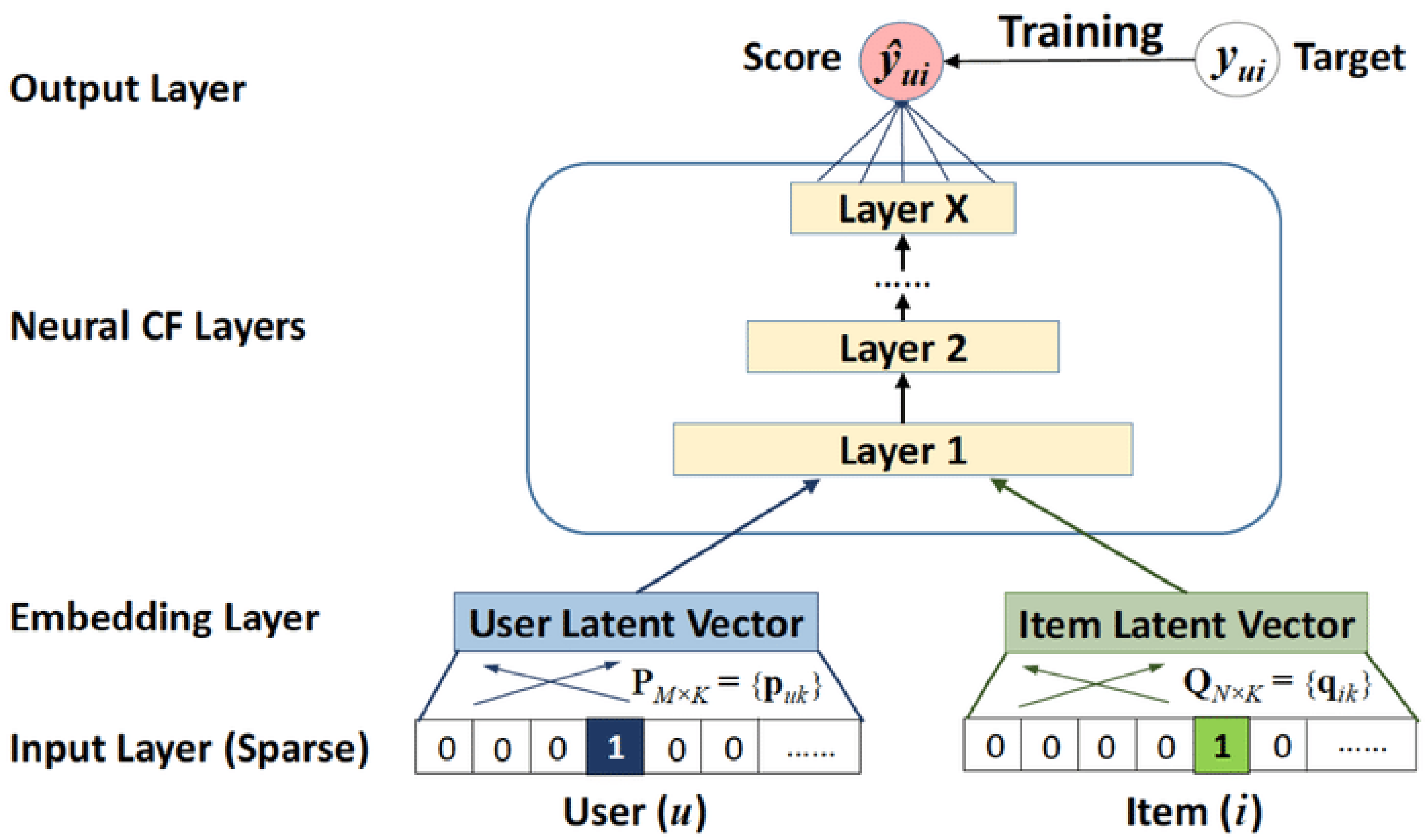
Multi-Layer Perceptron Experiments

BASELINE MODELS

Neural Factorization Machines (He et al., 2017)



Neural Collaborative Filtering (He et al., 2017)



MetaRec-MLP

BASE MODEL

Neural Collaborative Filtering

LOSS FUNCTION

Binary Cross-Entropy Loss

EVALUATION METRICS

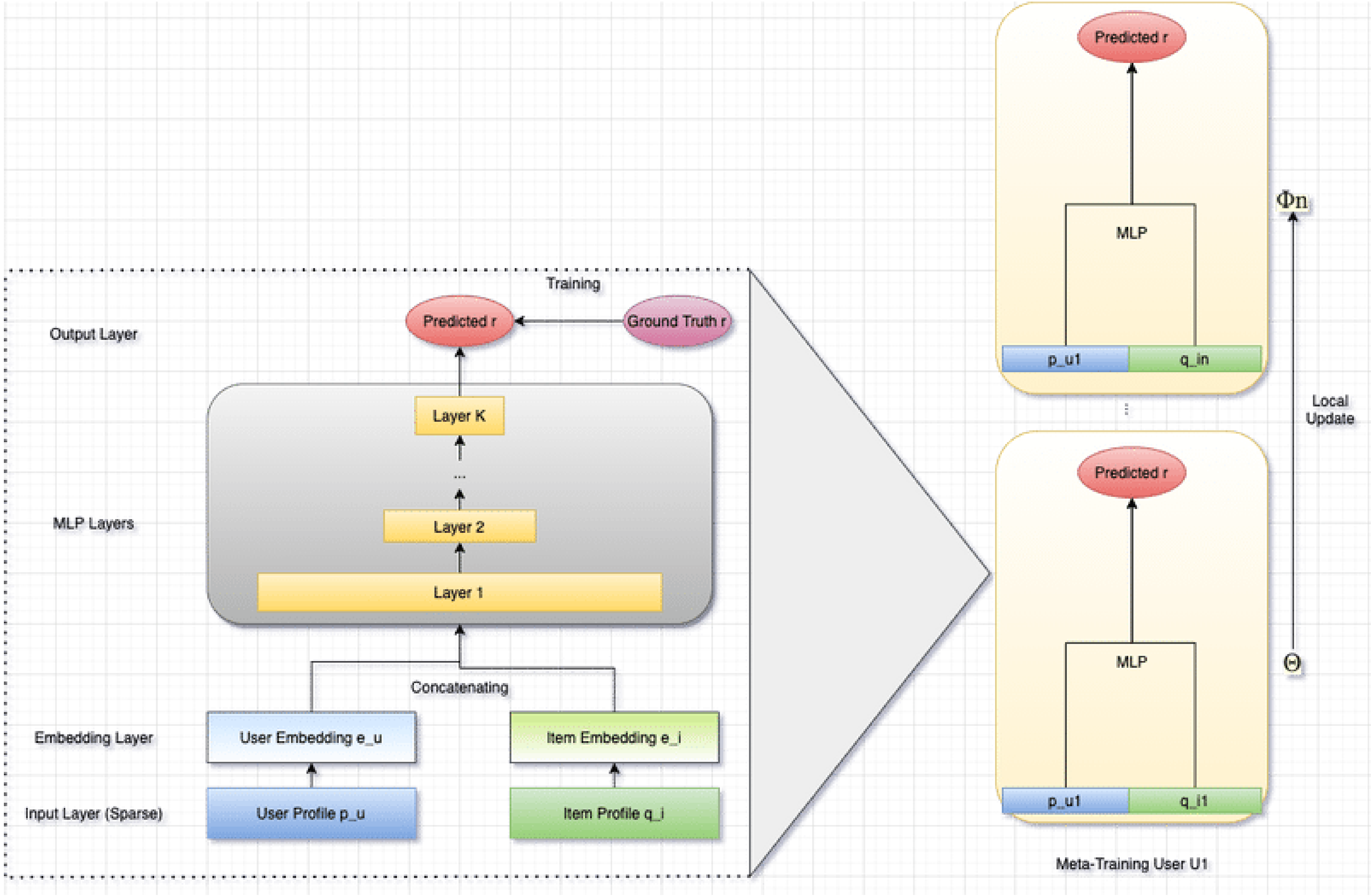
- Binary Classification with Implicit Feedback
- Area Under the ROC Curve (AUC)

$$e_u = F_{\theta_u}(p_u)$$
$$e_i = F_{\theta_i}(q_i)$$

$$\begin{aligned} x_0 &= [e_u, e_i], \\ x_1 &= \sigma(W_1 x_0 + b_1), \\ &\cdots \\ x_m &= \sigma(W_m x_{m-1} + b_m), \\ \hat{r}_{ui} &= \sigma(W_o x_m + b_o) \end{aligned}$$

$$L_u(F_{\theta,\phi_u}) = -\frac{1}{|B|}\sum_{i\in B}(r_{ui} log \hat{r}_{ui}) + (1 - r_{ui}) log(1 - \hat{r}_{ui})$$

ADAPTED FROM (HE ET AL., 2017)



Classification Results

Models	Test AUC	Validation AUC	Training Time
WideDeep	**0.7991**	**0.7995**	1h12m15s
DeepFM	0.7915	0.7918	1h10m50s
xDeepFM	0.7429	0.7408	2h15m17s
NeuralFM	0.7589	0.7560	1h36m0s
NeuralCF	0.7668	0.7673	**54m15s**
MetaRec-MLP (Ours)	**0.8135**	**0.8127**	**1h05m3s**

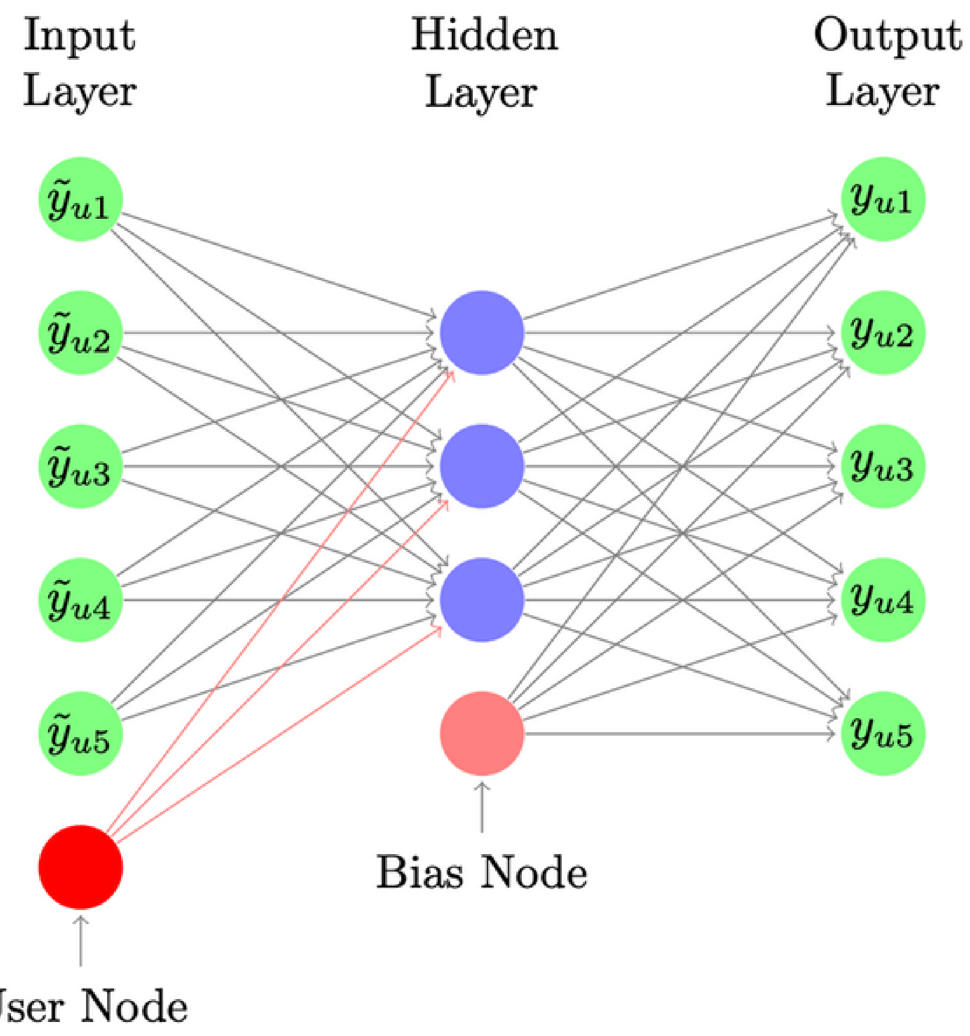
- Train-Valid-Test Split: 80-10-10
- Trained for 100 Epochs
- MetaRec-MLP outperforms other methods in AUC performance on both test and validation sets (Higher values are better)

- Tradeoff between model performance and compute cost
- WideDeep and DeepFM perform reasonably well
- xDeepFM overfits

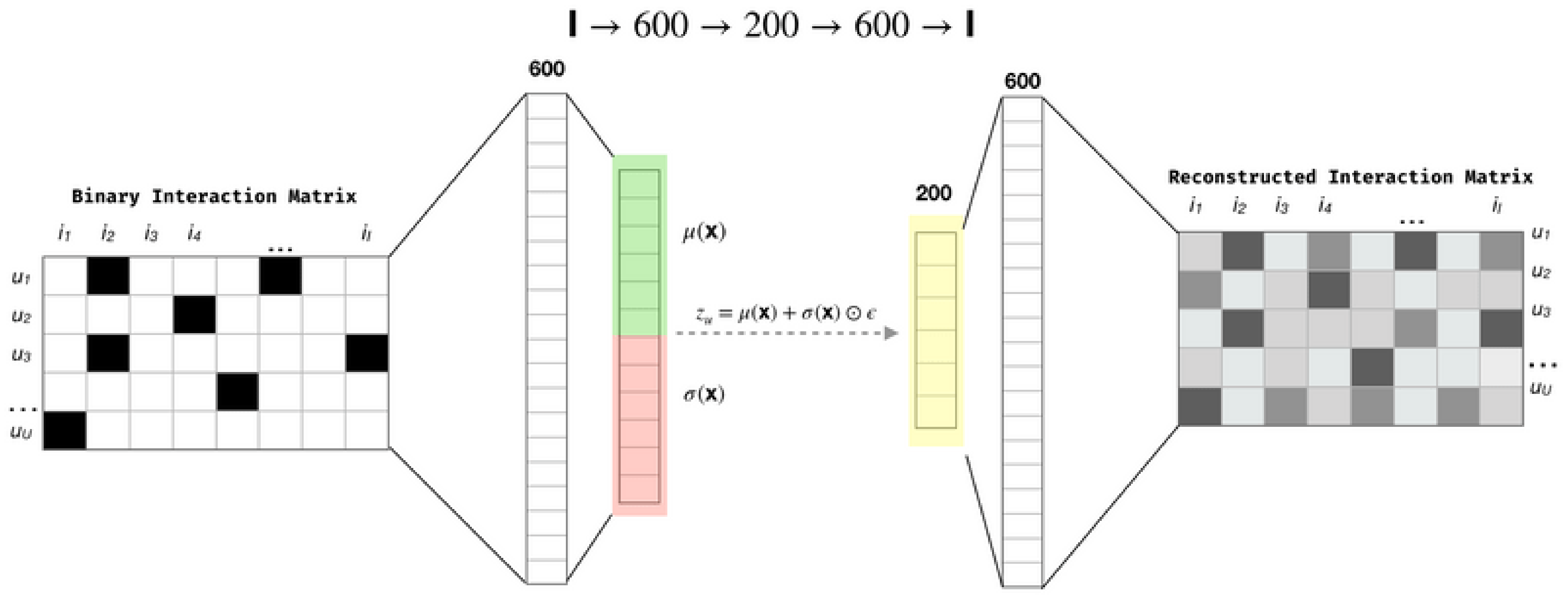
Autoencoder Experiments

BASELINE MODELS

Collaborative Denoising Autoencoder
(Wu et al., 2016)



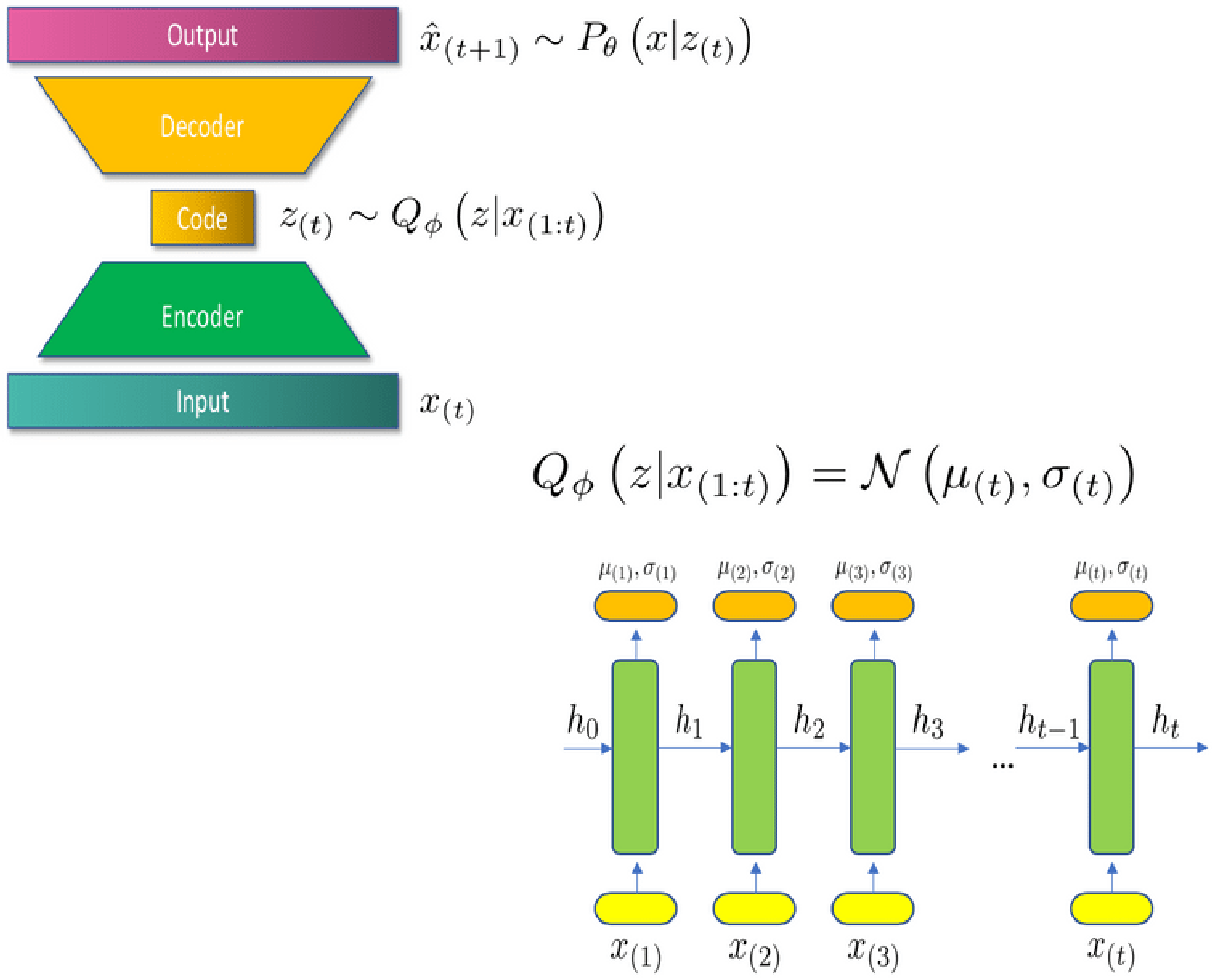
Multinomial Variational Autoencoder
(Liang et al., 2018)



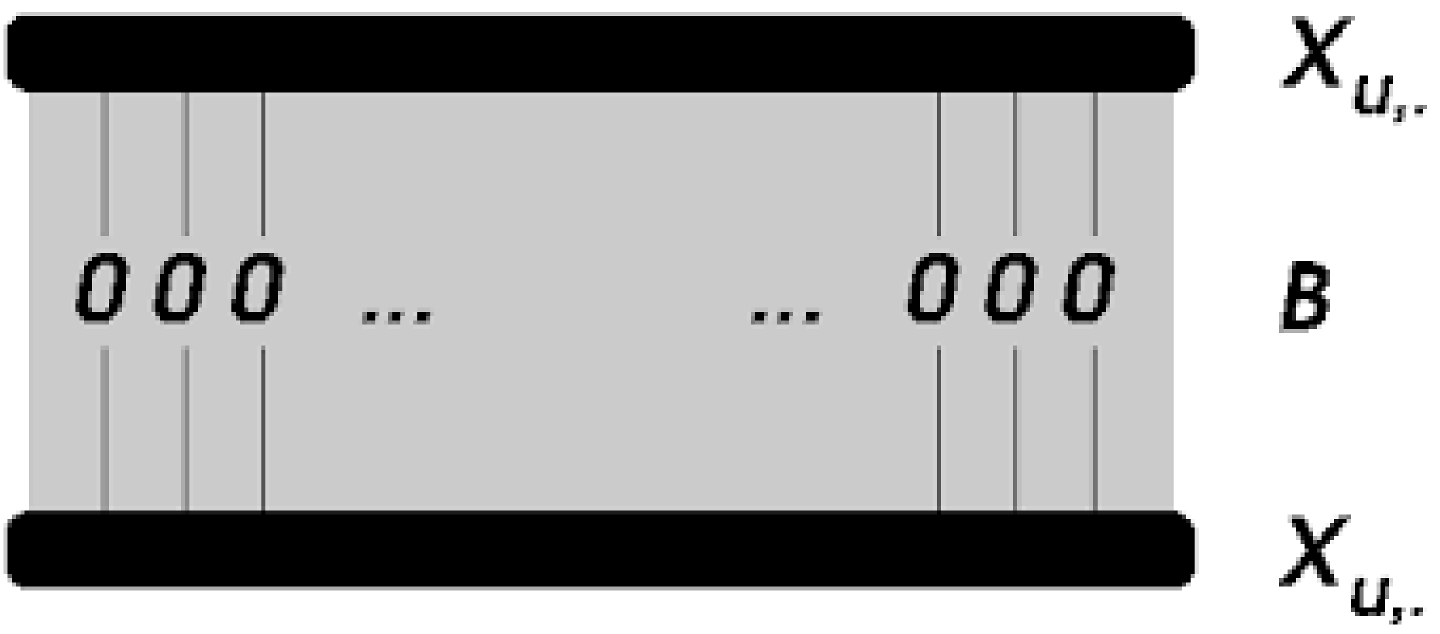
Autoencoder Experiments

BASELINE MODELS

Sequential Variational Autoencoder
(Sachdeva et al., 2019)



Embarrassingly Shallow Autoencoder
(Steck, 2019)



MetaRec-AE

$$\hat{r}_{ui} = (recon(r_{ui}, \hat{\theta}))$$

BASE MODEL

Vanilla Autoencoder

$$recon(r_{ui}; \theta) = f(W \cdot g(V r_{ui} + \mu) + b)$$

LOSS FUNCTION

Binary Cross-Entropy Loss with L2 Regularization

$$L_u(\theta) = -\frac{1}{m} \sum_{u=1}^{m} ||(r_{ui} log \hat{r}_{ui}) + (1 - r_{ui}) log(1 - \hat{r}_{ui})||_O^2 + \frac{\lambda}{2} \cdot (||W||_2^2 + ||V||_2^2)$$

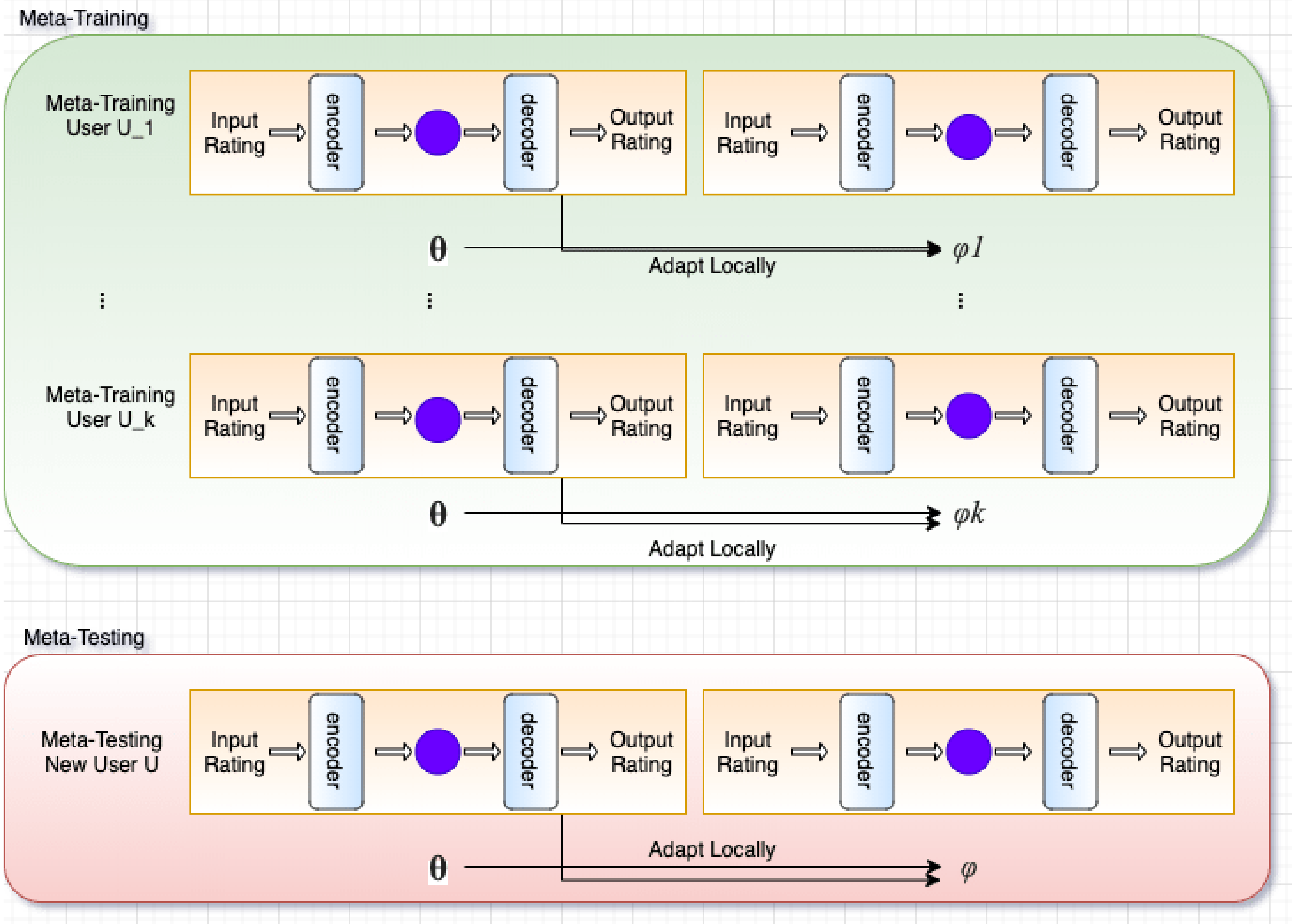
EVALUATION METRICS

- Top-K Recommendation with Implicit Feedback
- Precision
- Recall
- Normalized Discounted Cumulative Gain

$$Precision@k = \frac{Hits@k}{k}$$

$$Recall@k = \frac{Hits@k}{|R|}$$

$$NDCG@k = \frac{DCGs@k}{IDCG@k}$$


Meta-Training
Meta-Training User U_1
Input Rating
encoder
decoder
Output Rating
θ
Adapt Locally
φ1
Meta-Training User U_k
φk
Meta-Testing
Meta-Testing New User U
φ

Ranking Results

Models	Precision@100	Recall@100	NDCG@100	Training Time
CDAE	**8.94**	41.37	25.28	17m29s
MultVAE	**8.86**	41.15	25.08	**6m31s**
SVAE	8.18	**58.49**	**38.07**	6h37m19s
ESAE	7.57	41.81	25.61	**10m12s**
MetaRec-AE (Ours)	8.34	**55.12**	**33.06**	18m24s

- Train-Valid-Test Split: 80-10-10
- Trained for 100 epochs with early stopping
- MetaRec-AE had an average performance across three metrics (Higher values are better)
- SVAE was the best performing model in Recall and NDCG metrics
- MultVAE took the shortest time to train
- ESAE performed reasonably well

Conclusion

PART 6

Problem Formulation

RECOMMENDATION TASK

- Applications
- Challenges
- Collaborative Filtering
- Linear Models
- Neural Network Models

Proposed Framework

METAREC

- Model-Agnostic-Meta-Learning
- Leverages Learning From Previous Users To New Users
- Accurate: High Evaluation Metrics
- Rapid: In Small Number of Steps
- Efficient: Using Few Examples

Experimental Results

VS STATE-OF-THE-ARTS

- Matrix Factorization Models
- Multi-Layer Perceptron Models
- Auto-Encoder Models
- Explicit and Implicit Feedback
- Improved Accuracy Performance
- Computational Cost Tradeoff

Areas of Improvement

NOTES FOR FUTURE RESEARCH

Make Changes To The Base Model F(Θ)

- Complexify Them
- Simplify Them

Better MAML Training

- Training Instability
- Second-Order Derivative Cost
- Fixed Learning Rates During Updates

Areas of Improvement

NOTES FOR FUTURE RESEARCH

Other Meta-Learning Schemes For Recommendations

- Black-Box Meta-Learning
- Non-Parametric Meta-Learning

Addressing Scalability and Sparsity Challenges

- Shorter Training Time ~ Better Scalability
- Warm Users vs Cold Users Setup

References

- Chelsea Finn, Pieter Abbeel, and Sergey Levine. **Model-Agnostic Meta-Learning For Fast Adaptation Of Deep Networks**. In *International Conference on Machine Learning*, 2017.
- Maxwell Harper and Joseph Konstan. **The MovieLens Datasets: History and Context.** In *ACM Transactions on Interactive Intelligent Systems*, 2015.
- Yehuda Koren, Robert Bell, and Chris Volinsky. **Matrix Factorization Techniques For Recommender Systems.** In *IEEE Computer Society*, 2009.
- Steffen Rendle. **Factorization Machines.** In *IEEE Conference on Data Mining*, 2010.
- Maciej Kula. **Mixture-Of-Tastes Models For Representing Users With Diverse Interests.** 2017.
- Ian Porteous, Arthur Asuncion, and Max Welling. **Bayesian Matrix Factorization With Side Information and Dirichlet Process Mixtures.** 2010.
- Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, et al. **Wide and Deep Learning For Recommendation Systems.** In *RecSys*, 2016.
- Huifeng Guo, Ruiming Tang, Yunming Ye, et al. **DeepFM: A Factorization-Machine Based Neural Network For CTR Prediction.** 2017.
- Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, et al. **xDeepFM: Combining Explicit and Implicit Feature Interactions For Recommender Systems.** 2018.
- Xiangnan He and Tat-Seng Chua. **Neural Factorization Machines For Sparse Predictive Analytics.** 2017.
- Xiangnan He, Lizi Liao, Hangwang Zhang, et al. **Neural Collaborative Filtering.** In *The World Wide Web Conference*, 2017.
- Yao Wu, Christopher DuBois, Alice Zheng, and Martin Ester. **Collaborative Denoising Autoencoders For Top-N Recommender Systems.** In *Web Search and Data Mining*, 2016.
- Dawen Liang, Rahul Krishman, Matthew Hofman, and Tony Jebara. **Variational Autoencoders For Collaborative Filtering.** 2018.
- Noveen Sachdeva, Giuseppe Manco, Ettore Ritacco, and Vikram Pudi. **Sequential Variational Autoencoders For Collaborative Filtering.** In *Web Search and Data Mining*, 2019.
- Harld Steck. **Embarrassingly Shallow Autoencoders For Sparse Data.** In *The World Wide Web Conference*, 2019.

Thank You!

jl1165@rit.edu 😊

Codebase: github.com/khanhnamle1994/MetaRec

Appendix

Extra Details

MetaRec–MF

Algorithm 1: MAML Algorithm for MetaRec-MF

Input : Trainable global parameters θ, user set U, local and global hyper-parameters α, β

Output: Learned global parameters θ

Initialize θ randomly;
while *not converged* **do**
    Sample a batch of users B from U;
    for *user* $u \in B$ **do**
        Evaluate the gradient $\nabla_\theta L_u(F_\theta)$;
        Perform the local update $\phi_u \leftarrow \theta - \alpha \nabla_\theta L_u(F_\theta)$;
    end
    Perform global update $\theta \leftarrow \theta - \beta \sum_u L_u(F_{\phi_u})$
end

MetaRec–MLP

Algorithm 2: MAML Algorithm for MetaRec-MLP

Input : Trainable global parameters θ, user set U, local and global hyper-parameters α, β

Output: Learned global parameters θ

Initialize θ randomly;
Initialize ϕ randomly;
while *not converged* **do**
  Sample a batch of users B from U;
  for *user* $u \in B$ **do**
    Evaluate the gradient $\nabla_{\phi_u} L_u(F_{\theta,\phi_u})$;
    Perform the local update $\phi_u \leftarrow \phi_u - \alpha \nabla_{\phi_u} L_u(F_{\theta,\phi_u})$;
  end
  Perform global update $\theta \leftarrow \theta - \beta \sum_u \nabla_\theta L_u(F_{\theta,\phi_u})$;
  Perform global update $\phi \leftarrow \phi - \beta \sum_u \nabla_\phi L_u(F_{\theta,\phi_u})$
end

MetaRec–AE

Algorithm 4: MAML Algorithm for MetaRec-AE

Input : Trainable global parameters θ, user set U, local and global hyper-parameters α, β

Output: Learned global parameters θ

initialize $\theta^{(0)}$;
$i = 0$;
while *not converged* **do**
  for $u \in U$ **do**
    update autoencoder
    adapt locally: $\phi^{u} = \theta^{(0,i)} - \alpha \nabla_{\theta} L(\theta^{(0,i)})$
  end
  update globally: $\theta^{(0,i+1)} \leftarrow \theta^{(0,i)} - \beta \nabla_{\phi} L(\phi^{u})$
  $i \leftarrow i + 1$
end