Collaborative Metric Learning

Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie,Deborah Estrin

Index

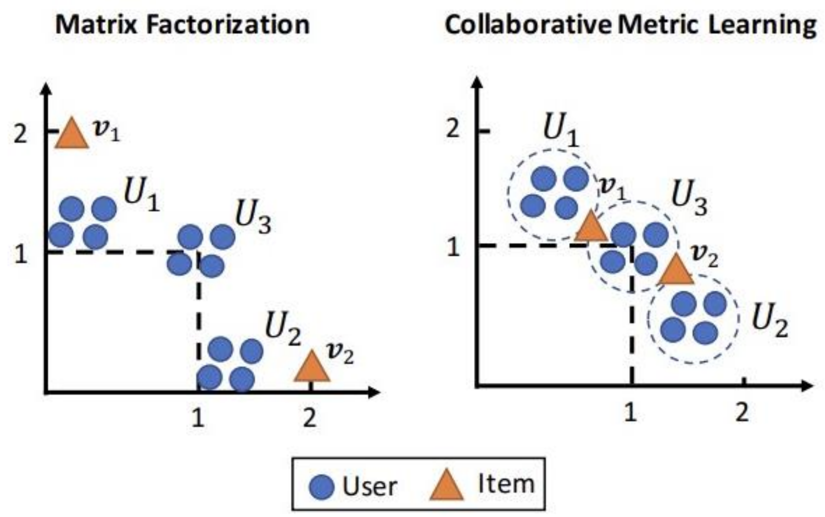
Introduction

- Distance is at the heart of many fundamental machine learning algorithms
  - Ex) K-nearest neighbor, K-means, SVMs

- Metric learning algorithms produce distance metric
  - similar & dissimilar
  - Assign smaller distances between similar pairs, larger distances between dissimilar pairs.

Introduction

- Triangle inequality
  - x is similar to y,z -> x pull two pairs closer, also pull the remaining pair (y,z) relatively close
  - matrix factorization ->does not satisfy the triangle inequality (dot product)



$$Matrix = \begin{bmatrix} 0 & 2 \\ 2 & 0 \\ 2 & 2 \end{bmatrix} \quad LatentUser = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \quad LatentItem = \begin{bmatrix} 0 & 2 \\ 2 & 0 \end{bmatrix}$$

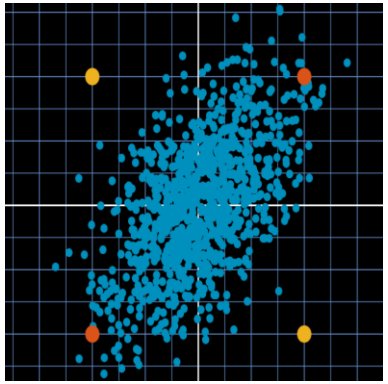
$$N \times M = (N \times K) \cdot (K \times M)$$

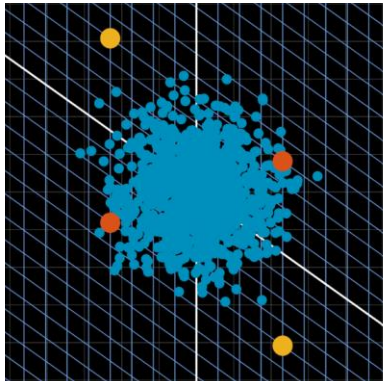
- what we want to know
  - users' preference
  - user-user & item-item similarity
    ->Collaborative Metric Learning
- explicit -> implicit

Background

- Metric Learning
  - $S = \{(x_i, x_j) | x_i \ and \ x_j \ are \ considered \ similar\}$
  - $D = \{(x_i, x_j) | x_i \ and \ x_j \ are \ considered \ dissimilar\}$

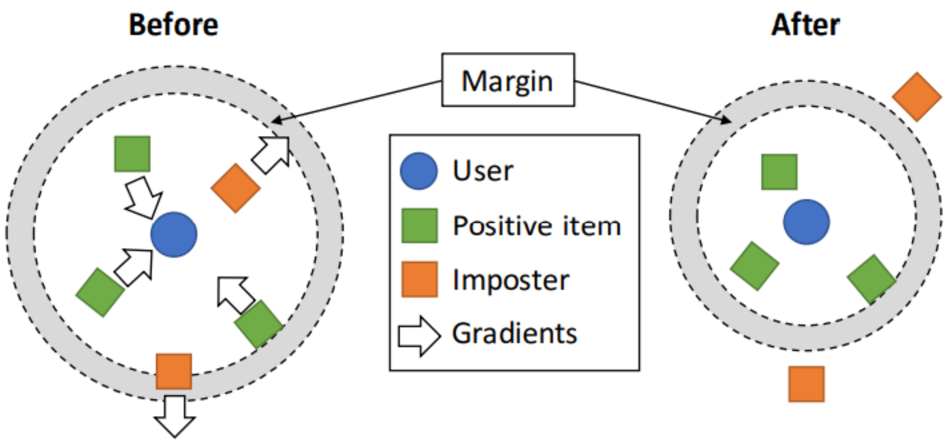
- Mahalanobis distance metric
  - $d_A(x_i, x_j) = \sqrt{(x_i - x_j)^T A (x_i - x_j)}$
    - where $A \in \mathbb{R}^{m \times m}$ is a positive semi-definite matrix





Background

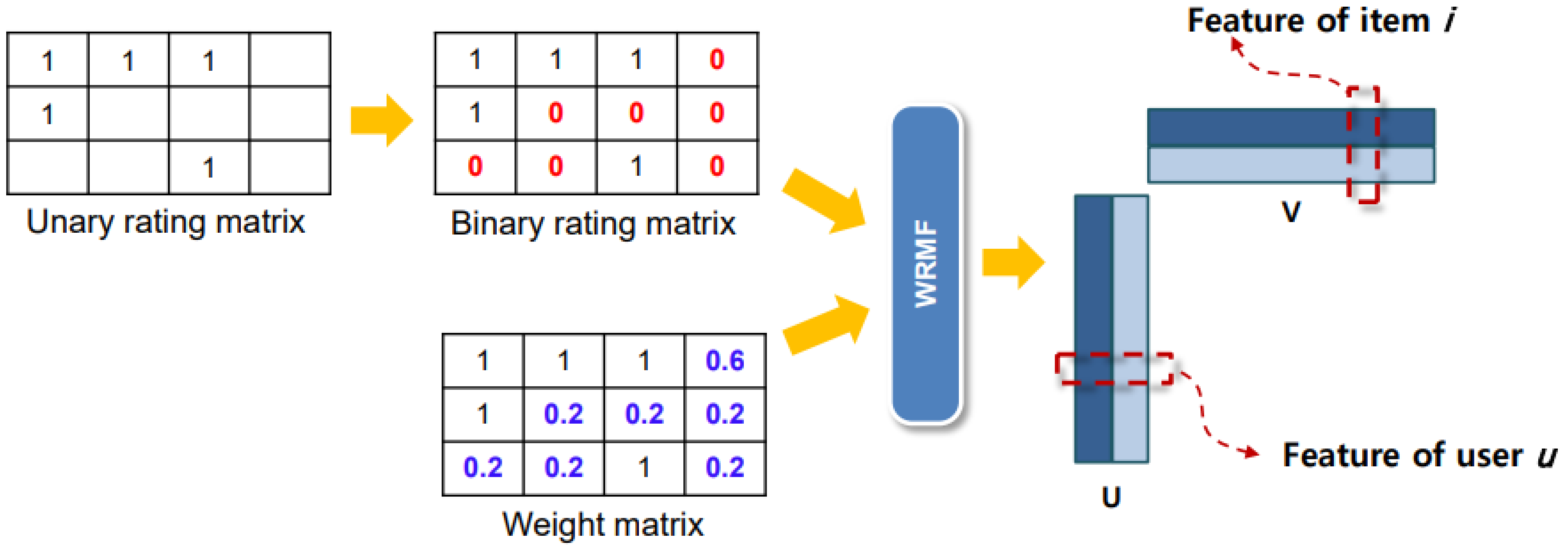
- Metric Learning for kNN
  - LMNN
    - $\mathcal{L}_{pull}(d) = \sum_{j \rightsquigarrow i} d\left(x_i, x_j\right)^2$
    - $\mathcal{L}_{push}(d) = \sum_{j \rightsquigarrow i} \sum_k (1 - y_{ik}) \left[1 + d\left(x_i, x_j\right)^2 - d(x_i, x_k)^2\right]_+$



Background

- Weighted regularized matrix factorization

$$\min_{\mathbf{u}_*,\mathbf{v}_*} \sum_{r_{ij}\in\mathcal{K}} c_{ij}(r_{ij} - \mathbf{u}_i^T\mathbf{v}_j)^2 + \lambda_u\|\mathbf{u}_i\|^2 + \lambda_v\|\mathbf{v}_i\|^2,$$



- Includes all the unobserved user-item interactions as negative samples
- Uses a case weight cij to reduce the impact of these uncertain samples

Background

- Bayesian Personalized Ranking

$$\min_{\mathbf{u}_*,\mathbf{v}_*} \sum_{i\in\mathcal{I}} \sum_{(j,k)\in\mathcal{D}_i} -log\ \sigma(\mathbf{u}_i^T\mathbf{v}_j - \mathbf{u}_i^T\mathbf{v}_k) + \lambda_u\|\mathbf{u}_i\|^2 + \lambda_v\|\mathbf{v}_j\|^2$$

  - (user,positive item,negative item)

Model description

- Model feature
  - Capture users' relative preferences for different items
  - Uses implicit data
  - Pulls the pairs in S closer and pushes the other pairs relatively further apart
  - By triangular inequality, will also cluster
    - User who co-like the same items together
    - Items that are co-liked by same users together

Model description

- Model formulation

$$min_{\theta,\mathbf{u}_*,\mathbf{v}_*} \mathcal{L}_m + \lambda_f \mathcal{L}_f + \lambda_f c_c$$

$$s.t. ||\mathbf{u}_*||^2 \leq 1 \text{ and } ||\mathbf{v}_*||^2 \leq 1$$

$\mathcal{L}_m$: Embedding Loss

$\mathcal{L}_f$: Feature Loss

$\mathcal{L}_c$: Covariance Loss

Model description

- Model formulation-Embedding Loss

$$\mathcal{L}_m(d) = \sum_{(i,j)\in S} \sum_{(i,k)\notin S} w_{ij}[m + D(i,j)^2 - d(i,k)^2]_+$$

  - $m + d(i,j)^2 > d(i,k)^2$ -> loss
    - m : margin
    - wij : weight

Model description

- Model formulation-Embedding Loss
  - Weighted Approximate-Rank Pairwise (WARP)
    - Penalize items at a lower rank

$$w_{ij} = log(rank_d(i,j) + 1)$$

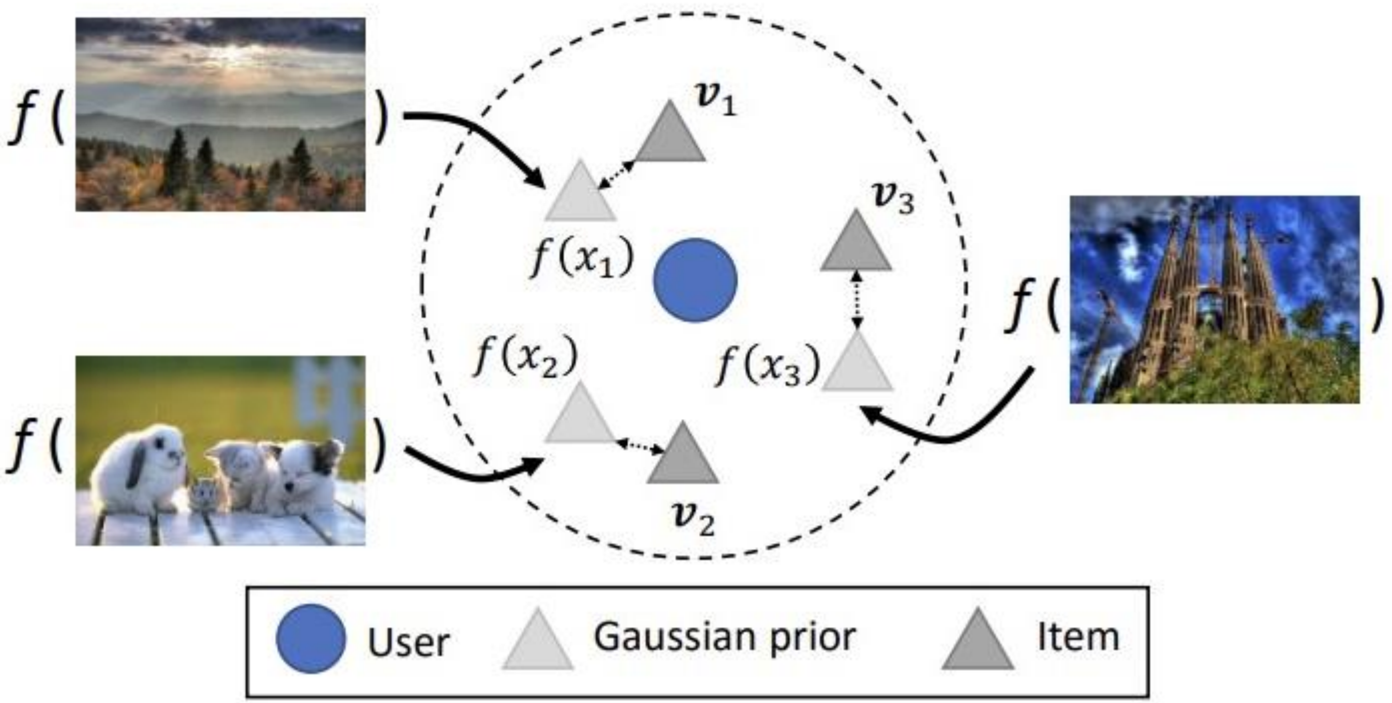
  - Sample U negative items in parallel and compute the hinge loss
  - Let MK denote the number of imposts in U sample, $rank_d(i,j)$ is approximated to $\left\lfloor \frac{J \times M}{U} \right\rfloor$

Model description

- Model formulation-feature loss

$$\mathcal{L}_f(\theta, \mathbf{v}_*) = \sum_j ||f(\mathbf{x}_j, \theta) - \mathbf{v}_j||^2$$

  - Think as transformation of input
  - x_j : raw item j vector
  - v_j : embedding item j vector
  - f(x) : MLP



Model description

- Model formulation-regularization
  - kNN based model is known to be ineffective in a high-dimensional space if the data points spread too widely

    -> bound all the user/item

    $\|u_*\| \leq 1$ and $\|v_*\| \leq 1$
- Covariance regularization

$$\mathcal{L}_c = \frac{1}{N}(\|C\|_f^2 - \|diag(C)\|_2^2)$$

$$C_{ij} = \frac{1}{N}\sum_n (y_i^n - \mu_i)(y_j^n - \mu_j)$$

Model Evaluation

- Dataset

Table 1: Dataset Statistics.

	CiteULike	BookCX	Flickr	Medium	MovieLens20M	EchoNest
Domain	Paper	Book	Photography	News	Movie	Song
# Users	7,947	22,816	43,758	61,909	129,797	766,882
# Items	25,975	43,765	100,000	80,234	20,709	260,417
# Ratings	142,794	623,405	1,372,621	2,047,908	9,939,873	7,261,443
Concentration[a]	33.47%	33.10%	13.48%	55.38%	72.52%	65.88%
Features Type	Tags	Subjects	Image Features	Tags	Genres, Keywords	NA
# Feature Dim.	10,399	7,923	2,048	2,313	10,399	NA

Model Evaluation

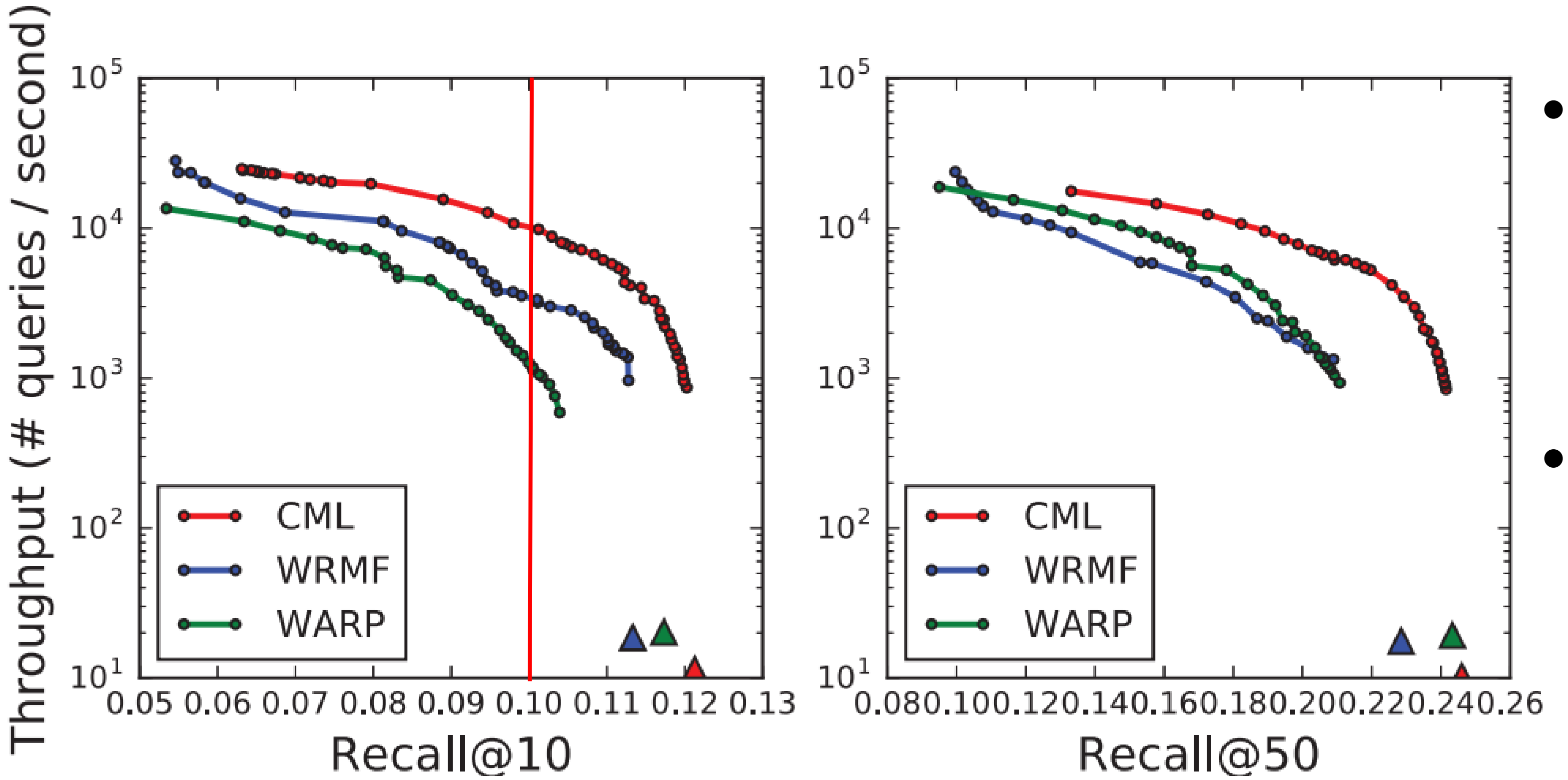
- Evaluation

Table 2: Recall@50 and Recall@100 on the test set. (# dimensions $r = 100$) The best performing method is boldfaced. $*$, $**$, $***$ indicate $p \leq 0.05$, $p \leq 0.01$, and $p \leq 0.001$ based on the Wilcoxon signed rank test suggested in [41].

	WRMF	BPR	WARP	CML	*ours vs. best*	FM	VBPR	CDL	CML+F	*ours vs. best*
Recall@50										
CiteULike	0.2437	0.2489	0.1916	**0.2714*** **	9.03%	0.1668	0.2807	**0.3375** **	0.3312	-1.86%
BookCX	0.0910	0.0812	0.0801	**0.1037*** **	13.95%	0.1016	0.1004	0.0984	**0.1147*** **	12.89%
Flickr	0.0667	0.0496	0.0576	**0.0711*** **	6.59%	NA	0.0612	0.0679	**0.0753*** **	10.89%
Medium	0.1457	0.1407	0.1619	**0.1730*** **	6.41%	0.1298	0.1656	0.1682	**0.1780*** **	5.82%
MovieLens	0.4317	0.3236	0.4649	**0.4665**	0.34%	0.4384	0.4521	0.4573	**0.4617***	0.96%
EchoNest	0.2285	0.1246	0.2433	**0.2460**	1.10%	NA	NA	NA	NA	NA
Recall@100										
CiteULike	0.3112	0.3296	0.2526	**0.3411*** **	3.37%	0.2166	0.3437	0.4173	**0.4255** **	1.96%
BookCX	0.1286	0.1230	0.1227	**0.1436*** **	11.66%	0.1440	0.1455	0.1428	**0.1712*** **	17.66%
Flickr	0.0821	0.0790	0.0797	**0.0922*** **	12.30%	NA	0.0880	0.0909	**0.1048*** **	15.29%
Medium	0.2112	0.2078	0.2336	**0.2480*** **	6.16%	0.1900	0.2349	0.2408	**0.2531*** **	5.10%
MovieLens	0.5649	0.4455	0.5989	**0.6022**	0.55%	0.5561	0.5712	0.5943	**0.5976**	0.55%
EchoNest	0.2891	0.1655	0.3021	**0.3022**	0.00%	NA	NA	NA	NA	NA

Model Evaluation

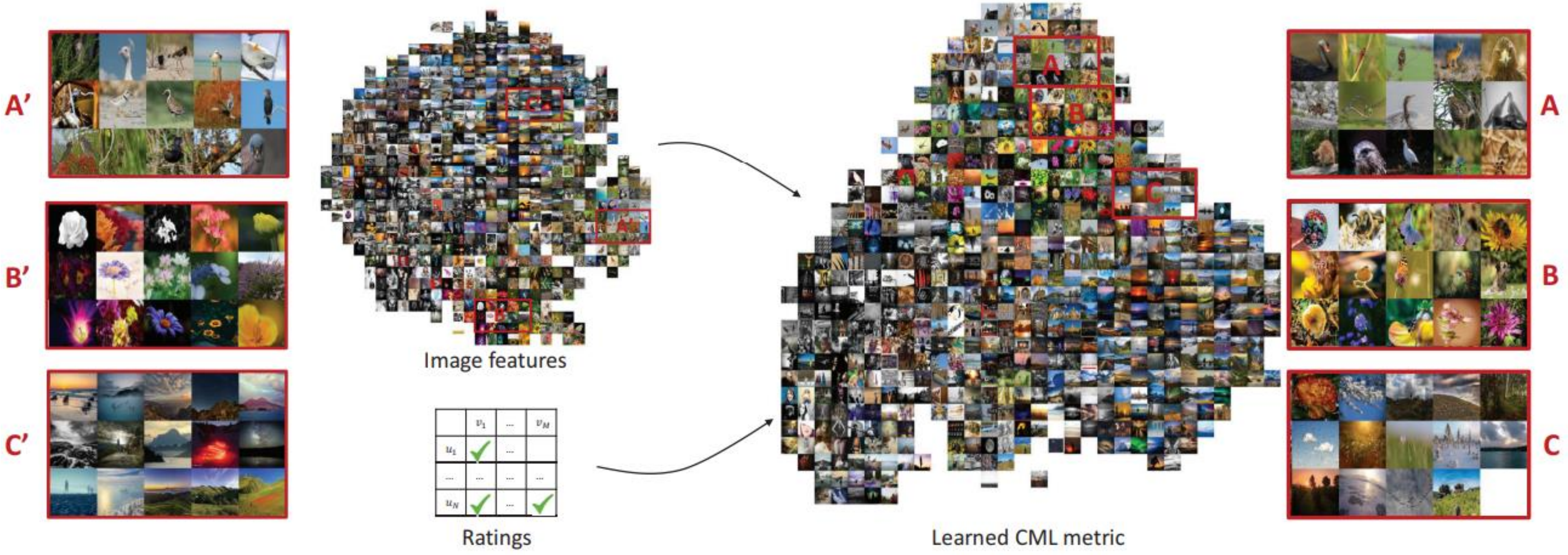
- Evaluation



- CML's throughput is improved by 106x with only 2% reduction in accuracy
- Over 8x faster than (optimized) MF models given the same accuracy

Model Evaluation

- Evaluation



Conclusion

- Conclusion

Task	Dataset	Model	Metric Name	Metric Value	Global Rank	Benchmark
Recommendation Systems	Million Song Dataset	CML	Recall@50	0.2460	# 6	Compare
			Recall@100	0.3022	# 1	Compare
Recommendation Systems	MovieLens 1M	CML	HR@10	0.7216	# 6	Compare
			nDCG@10	0.5413	# 5	Compare
Recommendation Systems	MovieLens 20M	CML	Recall@50	0.4665	# 9	Compare
			HR@10	0.7764	# 3	Compare
			nDCG@10	0.5301	# 3	Compare
Recommendation Systems	Netflix	CML	nDCG@10	0.2948	# 3	Compare
			Recall@10	0.4612	# 2	Compare