Scaling up Link Prediction with Ensembles

Liang Duan[1], Charu Aggarwal[2], Shuai Ma[1], Renjun Hu[1], Jinpeng Huai[1]

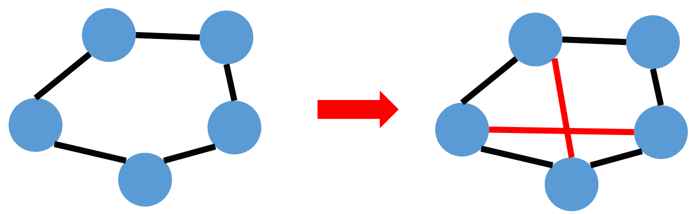
[1]SKLSDE Lab, Beihang University, China

[2]IBM T. J. Watson Research Center, USA

Motivation

- **Link prediction**

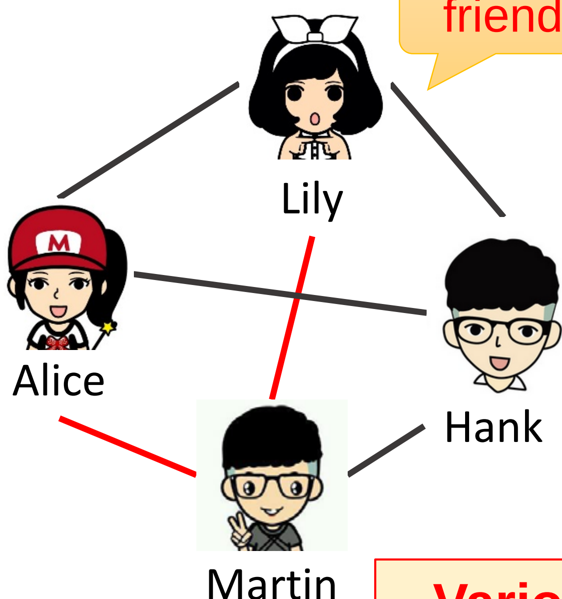
  predicting the formation of future links in a dynamic network



- **Applications**

  recommender systems, examples:

friends



movies

collaborators

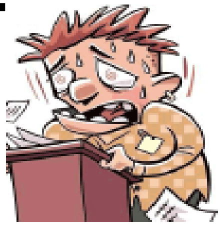
Various applications in large networks!

Motivation

- **The $O(n^2)$ problem in link prediction**
  - ✓ Assume a node pair could be done in a single machine cycle.
  - ✓ A network with *n* nodes contains $O(n^2)$ possible links.
  - ✓ Analysis of required time:

Network Sizes	1 GHz	3 GHz	10 GHz
10^6 nodes	1000 sec.	333 sec.	100 sec.
10^7 nodes	27.8 hrs	9.3 hrs	2.78 hrs
10^8 nodes	> 100 days	> 35 days	> 10 days
10^9 nodes	> 10000 days	> 3500 days	> 1000 days

It is **challenging** to search the entire space in large networks!



Most existing methods only search over a subset of possible links rather than the entire network.

Outline

- Latent Factor Model for Link Prediction
- Structural Bagging Methods
- Experimental Study
- Summary

Latent Factor Model for Link Prediction

- **Network *G(N, A)* and weight matrix *W***

  G: an undirected graph
  N: node set of *G* containing *n* nodes
  A: edge set of *G* containing *m* edges
  W: an $n \times n$ matrix containing the weights of the edges in *A*

- **Nonnegative Matrix factorization (NMF)** $W \approx FF^T$

  ✓ F_i is an *r*-dimensional latent factor with the *i*-th node.

  ✓ determine *F* by $\min_{F \geq 0} \| W - FF^T \|^2$ using multiplicative update rule:

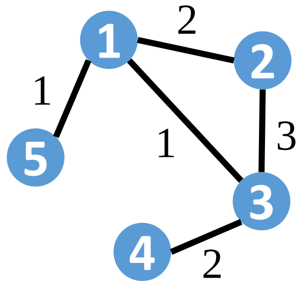
$$F_{ij} \leftarrow F_{ij}\left(1 - \beta + \beta \frac{(WF)_{ij}}{(FF^T F)_{ij}}\right), \beta \in (0,1]$$

- **Link prediction**

  positive entries in FF^T are viewed as predictions of 0-entries in *W*

Latent Factor Model for Link Prediction

- **Example 1:** Given a network with 5 nodes and r = 3, predict links on this network.

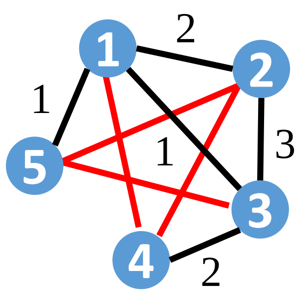


$$W = \begin{bmatrix} 0 & 2 & 1 & 0 & 1 \\ 2 & 0 & 3 & 0 & 0 \\ 1 & 3 & 0 & 2 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$F = \begin{bmatrix} 0.7 & 0.3 & 0.7 \\ 0.5 & 0.7 & 0.9 \\ 0.4 & 1.1 & 0.7 \\ 0 & 0.8 & 0 \\ 0.5 & 0 & 0.1 \end{bmatrix}$$

$$F^T = \begin{bmatrix} 0.7 & 0.5 & 0.4 & 0 & 0.5 \\ 0.3 & 0.7 & 1.1 & 0.8 & 0 \\ 0.7 & 0.9 & 0.7 & 0 & 0.1 \end{bmatrix}$$

$$FF^T = \begin{bmatrix} 0 & 2 & 1 & 0.3 & 1 \\ 2 & 0 & 3 & 0.6 & 0.3 \\ 1 & 3 & 0 & 2 & 0.2 \\ 0.3 & 0.6 & 2 & 0 & 0 \\ 1 & 0.3 & 0.2 & 0 & 0 \end{bmatrix}$$



Latent Factor Model for Link Prediction

- **Efficient top-*k* prediction searching is necessary**

  FF^T contains n^2 entries

  F is often nonnegative and sparse

- **Top-(ε, *k*) prediction problem is to return *k* predicted links**

  the k-th best value of FF^T for a link (i, j) is at most ε less than the k-th best value of FF^T over any link (h, l) in the network.

  A tolerance of ε helps in speeding up the search process

Top-(ε, k) Prediction Searching Method

- **A solution for top-(ε, k) prediction problem**

Execute the following nested loop for each column of S:

f_p (f_p'): the number of rows in the p-th column of S that $S_{ip} > \sqrt{\varepsilon / r}$ (0)

for each $i = 1$ to f_p **do**

  for each $j = i + 1$ to f_p' **do**

    if $S_{ip} \cdot S_{jp} < \varepsilon / r$ **then** break inner loop;

    else increase the score of node-pair (R_{ip}, R_{jp}) by an amount of $S_{ip} \cdot S_{jp}$;

  end for

end for

S: sorting the columns of F in a descending order

R: node identifiers of F arranged according to the sorted order of S

outer loop	$S_{ip} < \sqrt{\varepsilon / r}$
inner loop	$S_{ip} \cdot S_{jp} < \varepsilon / r$

→ underestimation is at most ε

Top-(ε, k) Prediction Searching Method

- Example: Continue with example 1, assume $\varepsilon = 1$.

$$\begin{bmatrix} 0.7 & 0.3 & 0.7 \\ 0.5 & 0.7 & 0.9 \\ 0.4 & 1.1 & 0.7 \\ 0 & 0.8 & 0 \\ 0.5 & 0 & 0.1 \end{bmatrix} \rightarrow \begin{bmatrix} 0.7 & 1.1 & 0.9 \\ 0.5 & 0.8 & 0.7 \\ 0.5 & 0.7 & 0.7 \\ 0.4 & 0.3 & 0.1 \\ 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 3 & 2 \\ 2 & 4 & 1 \\ 5 & 2 & 3 \\ 3 & 1 & 5 \\ 4 & 5 & 4 \end{bmatrix}$$

F S R

$$\sqrt{\varepsilon / r} \approx 0.58$$

$$\varepsilon / r \approx 0.33$$

Column 1: $f_1 = 1$, $f_1' = 4$, $S_{11}*S_{21} = 0.35$, $S_{11}*S_{31} = 0.35$,

Column 2: $f_2 = 3$, $f_2' = 4$, $S_{12}*S_{22} = 0.88$, $S_{12}*S_{32} = 0.77$, $S_{12}*S_{42} = 0.33$, $S_{22}*S_{32} = 0.56$

Column 3: $f_3 = 3$, $f_3' = 4$, $S_{13}*S_{23} = 0.63$, $S_{13}*S_{33} = 0.63$, $S_{23}*S_{33} = 0.49$

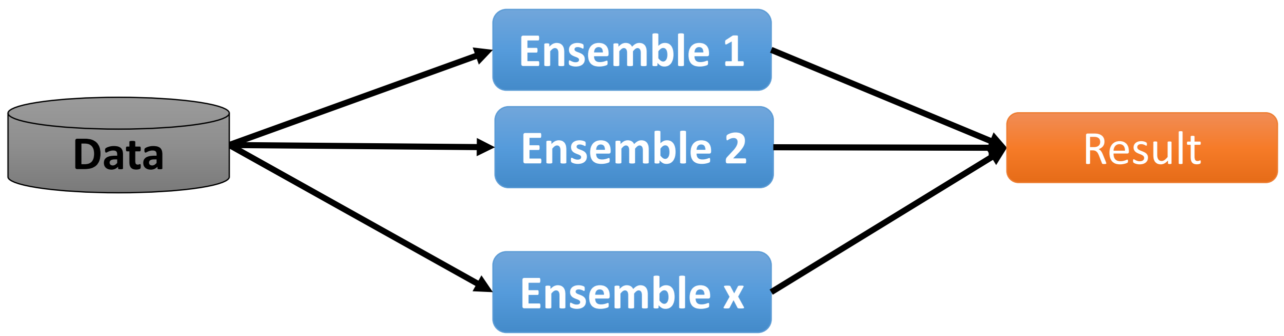
A large portion of search space is pruned!

Outline

- Latent Factor Model for Link Prediction
- Structural Bagging Methods
- Experimental Study
- Summary

Structural Bagging Methods

- **Problems in latent factor models**
  - ✓ the complexity is $O(nr^2)$
  - ✓ r usually increases with the network size
  - ✓ bad performance (efficiency & accuracy) on large sparse networks
- **Structural bagging methods**



  - ✓ decompose the link prediction problem into smaller sub-problems
  - ✓ aggregate results of multiple ensembles to a robust result

ensemble-enabled method

- **Efficiency advantages**
  - ✓ smaller sizes of the matrices in NMF
  - ✓ smaller the number r of latent factors

Random Node Bagging

- **Steps:**

f : fraction of the number of nodes to be selected

1. $N_r \leftarrow f \times n$ nodes selected randomly from G

   $N_s \leftarrow N_r \bigcup \{\text{nodes adjacent to } N_r\}$

2. $W_s \leftarrow$ weight matrix of subgraph induced on N_s of G

3. $F_s \leftarrow$ factorization of W_s by NMF

   $R \leftarrow$ top-(ε, k) on F_s // R is the set of predictions

- **Bound of random node bagging**

  The expected times of each node pair included in μ / f^2 ensembles is at least μ.

Edge & Biased Edge Bagging

Random node bagging samples less relevant regions.

- **Edge bagging**

Steps:

1. $N_s \leftarrow$ a single node selected randomly from G

   while $|N_s| < f \times n$ do

   $N_t \leftarrow \{\text{nodes adjacent to } N_s\}$

   if $|N_t|$ then $N_s \leftarrow N_s \bigcup \{\text{a single node selected randomly from } N_t\}$

   else $N_s \leftarrow N_s \bigcup \{\text{a single node selected randomly from } G\}$

Steps 2 and 3 are same to the random node bagging.

Edge bagging tends to include high degree nodes.

- **Biased edge bagging**

Difference with edge bagging:

$$\text{if } |N_t| \text{ then } N_s \leftarrow N_s \bigcup \{\text{the node with the least sampled times in } N_t\}$$

Using Link Prediction Characteristics

- Bagging should be designed in particular for link prediction.

Observation

Most of all new links span within short distances (closing triangles)

- **Combine link prediction characteristics**

a node should be always sampled together with all its neighbors.

- **Example:**

✓ The edge (*c*, *d*) is a triangle-closing edge.

✓ When the node *a* is selected, its neighbors *b*, *c*, *d* and *e* are also put into the same ensemble.

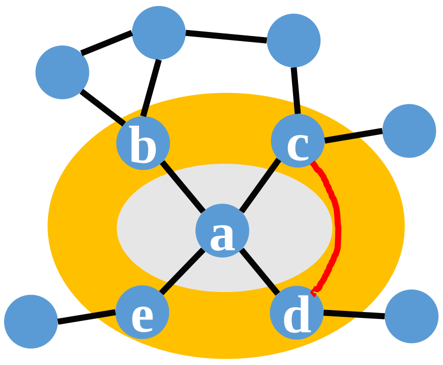


Figure 1: Triangle-closing model.

Ensemble Enabled Top-k Predictions

- **Framework for ensemble-enabled top-k prediction**

a network $G(N, A)$ and parameters μ and f

repeat μ / f^2 times **do**

1: $N_s \leftarrow$ ensemble generated by one of node, edge and biased edge bagging;

2: Compute F_s by factorizing W_s using NMF;

3: Obtain Γ' using top-(ε, k) method on F_s;

4: $\Gamma \leftarrow$ top-k largest value node pairs in $\Gamma' \cup \Gamma$;

maximum value

return Γ

Outline

- Latent Factor Model for Link Prediction
- Structural Bagging Methods
- Experimental Study
- Summary

Experimental Settings

- **Datasets**:

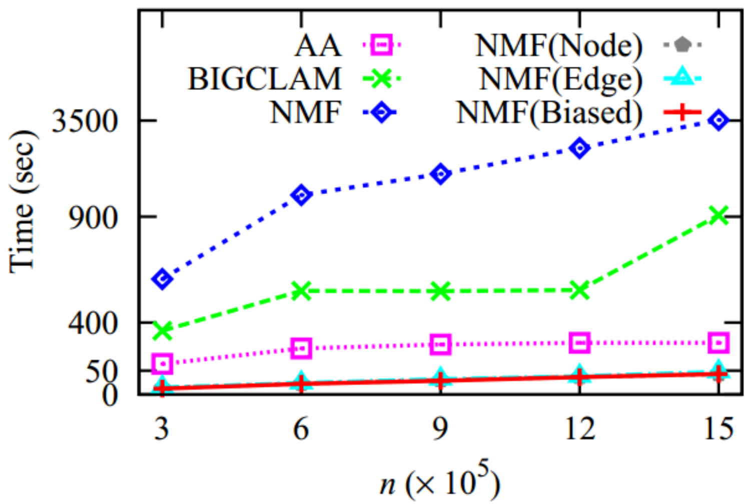
Datasets	Descriptions	# of nodes	# of edges
YouTube	friendship	3,223,589	9,375,374
Flickr	friendship	2,302,925	33,140,017
Wikipedia	hyperlink	1,870,709	39,953,145
Twitter	follower	41,652,230	1,468,365,182
Friendster	friendship	68,349,466	2,586,147,869

- **Algorithms**:
  - ✓ AA — the popular neighborhood based method Adamic/Adar
  - ✓ BIGCLAM — a probabilistic generative model based on community affiliations
  - ✓ NMF — our latent factor model for link prediction
  - ✓ NMF(Node) — NMF with random node bagging
  - ✓ NMF(Edge) — NMF with edge bagging
  - ✓ NMF(Biased) — NMF with biased edge bagging

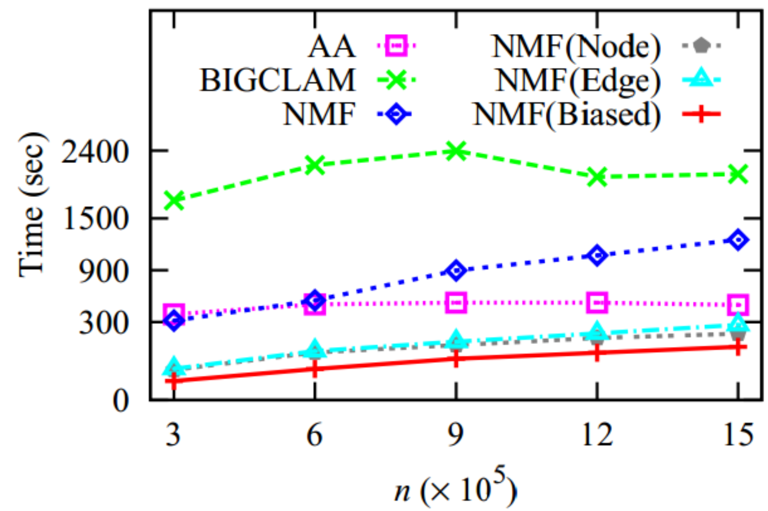
- **Implementation**:
  - ✓ All algorithms were written in C/C++ with no parallelization
  - ✓ 2 Intel Xeon 2.4GHz CPUs and 64GB of Memory

Efficiency Test

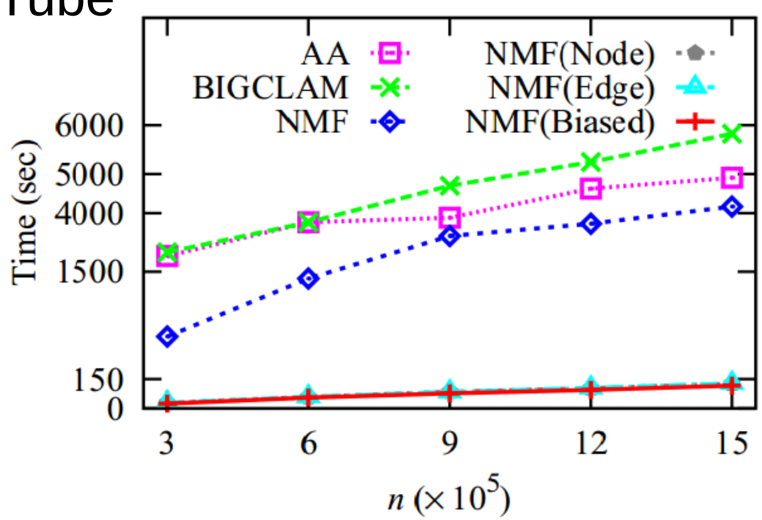
Efficiency comparison: with respect to the network sizes.



(a) YouTube



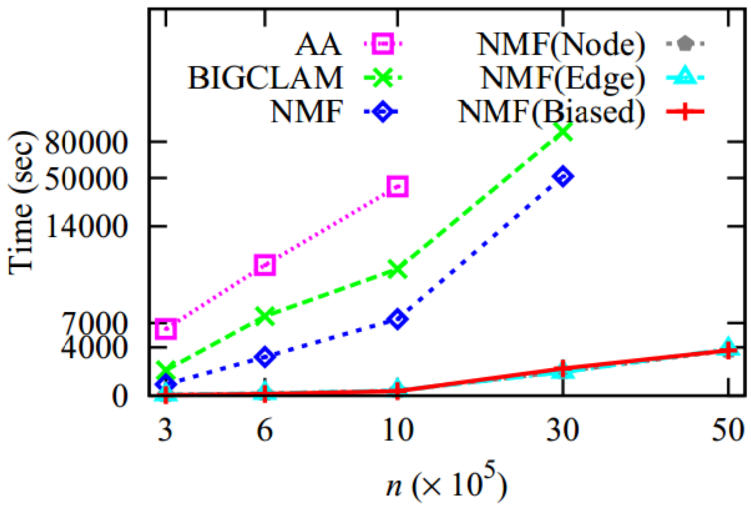
(b) Flickr



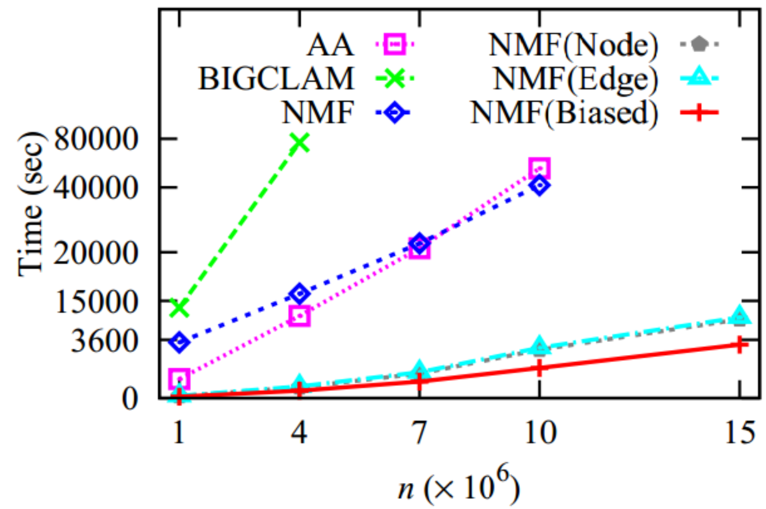
(c) Wikipedia

Efficiency Test

Efficiency comparison: with respect to the network sizes.



(d) Twitter



(e) Friendster

Dataset	NMF	AA	BIGCLAM
Twitter	20x	107x	43x
Friendster	31x	21x	175x

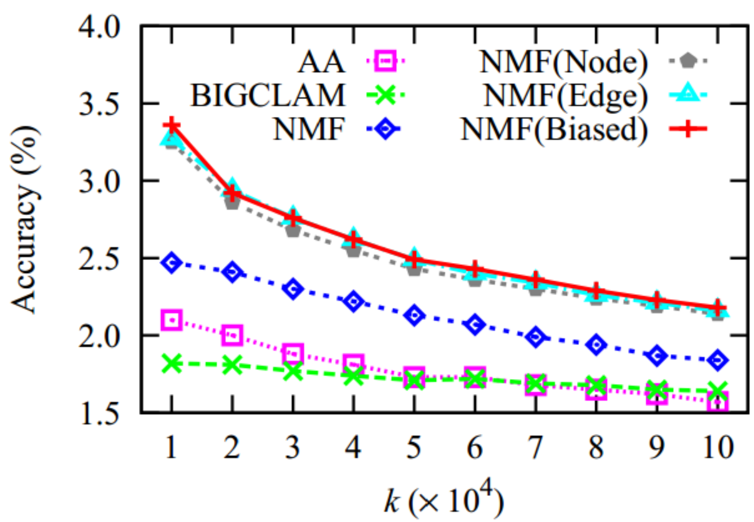
Table 2: The speedup of NMF(Biased) compared with other methods.

Effectiveness Test

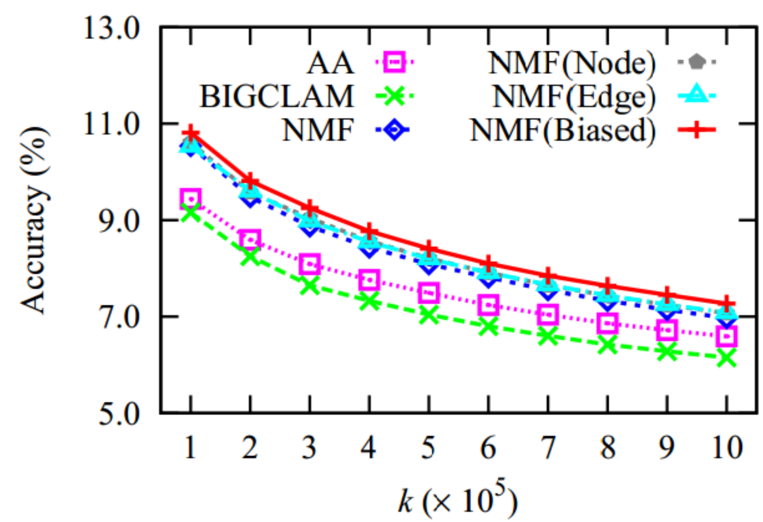
The effectiveness of a top-*k* link prediction method *x* is evaluated with the following measure:

$$accuracy(x) = \frac{\text{\# of correctly predicted links}}{\text{the number } k \text{ of predicted links}}$$

Accuracy comparison: with respect to the number *k* of predicted links.



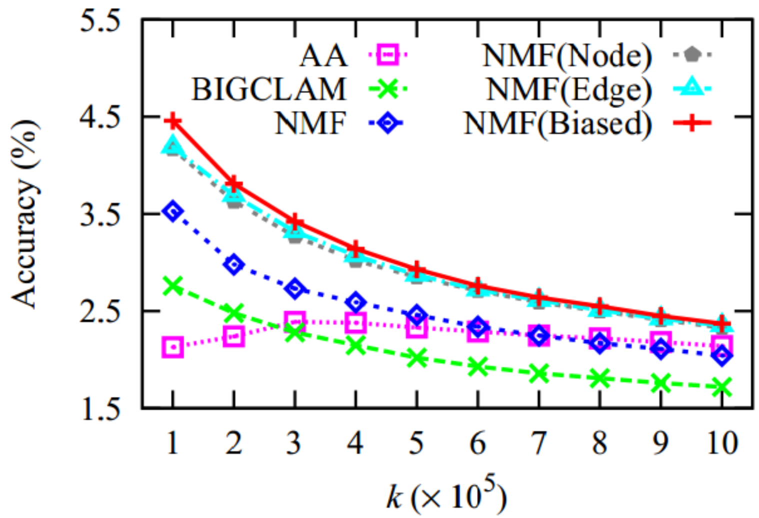
(a) YouTube



(b) Flickr

Effectiveness Test

Accuracy comparison: with respect to the number *k* of predicted links.



(c) Wikipedia

Dataset	NMF	AA	BIGCLAM
YouTube	18%	39%	33%
Flickr	4%	10%	18%
Wikipedia	16%	11%	38%

Table 2: The accuracy improved by NMF(Biased) compared with other methods.

Both efficiency and accuracy are improved!

Outline

- Latent Factor Model for Link Prediction
- Structural Bagging Methods
- Experimental Study
- Summary

Summary

- **Conclusions**
  - ✓ an ensemble-enabled approach for top-k link prediction;
  - ✓ scale to large networks with over 15 million nodes and 1 billion edges;
  - ✓ both accuracy and efficiency improved.

Accuracy and efficiency improved by NMF(Biased) compared with NMF:

Dataset	Accuracy	Dataset	Speedup
YouTube	18%	Twitter	20x
Flickr	4%		
Wikipedia	16%	Friendster	31x

- **Future work**
  - ✓ distributed approaches scalable on networks with billions of nodes;
  - ✓ personalized recommendation using our approach.

Thanks!

Q & A

Experimental Settings

- Training and ground truth data

Datasets	Date	# of nodes	# of edges
YouTube	2006-12-09 — 2007-02-22	1,503,841	3,691,893
	2007-02-23 — 2007-07-22	1,503,841	806,213
Flickr	2006-11-01 — 2006-11-30	1,580,291	13,341,698
	2006-12-01 — 2007-05-17	1,580,291	3,942,599
Wikipedia	2001-02-19 — 2006-10-31	1,682,759	28,100,011
	2006-11-01 — 2007-04-05	1,682,759	5,856,896

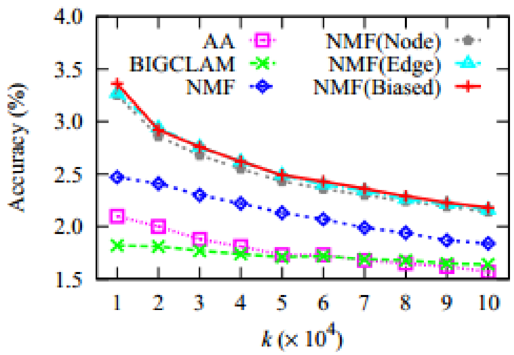
The data in the first time slot is the training data and the remaining is the ground truth data.

The latest five month part is treated as its ground truth.

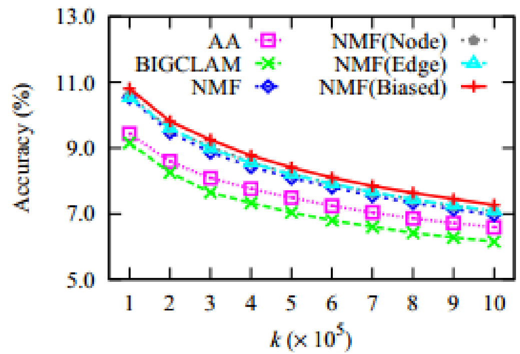
Twitter and Friendster do not have timestamps and are only used for the scalability test.

Experimental Results

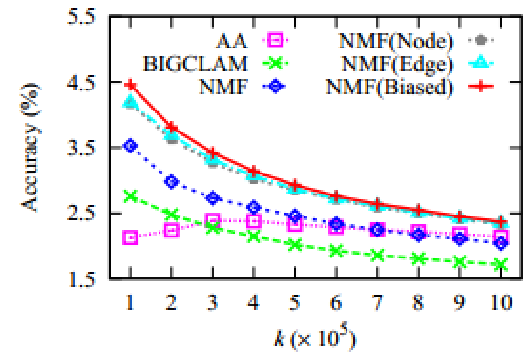
Accuracy and efficiency comparison: with respect to the number *k* of predicted links



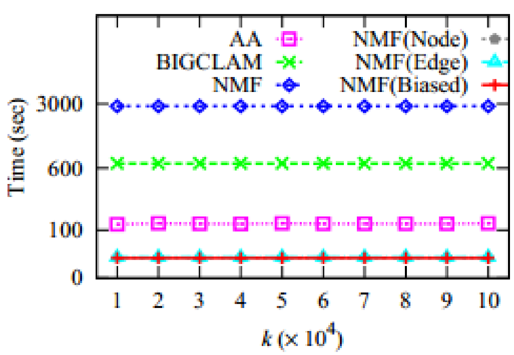
(a) YouTube



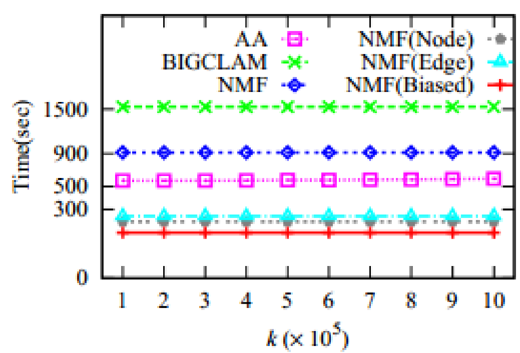
(b) Flickr



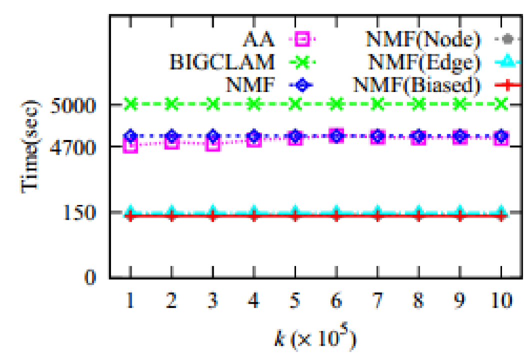
(c) Wikipedia



(d) YouTube



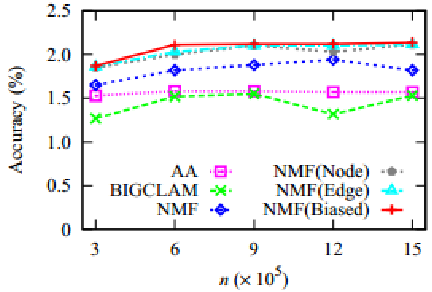
(e) Flickr



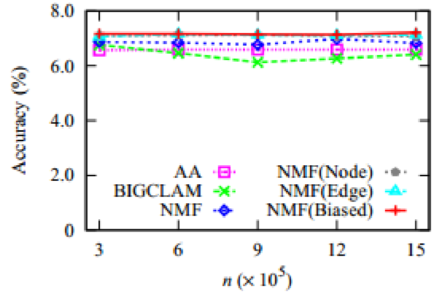
(f) Wikipedia

Experimental Results

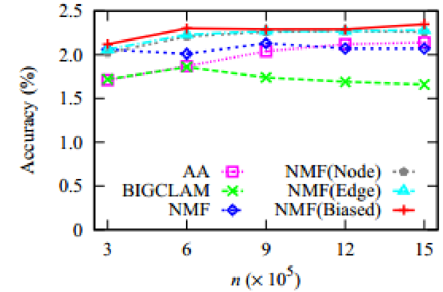
Accuracy and efficiency comparison: with respect to the network sizes



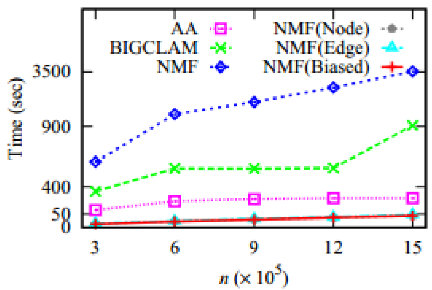
(a) YouTube



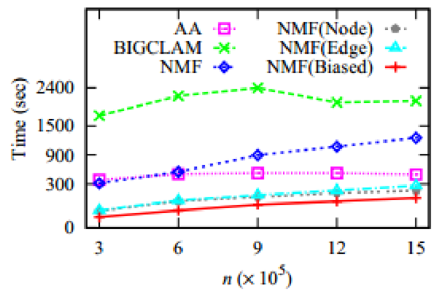
(b) Flickr



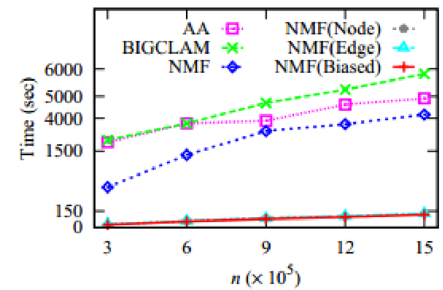
(c) Wikipedia



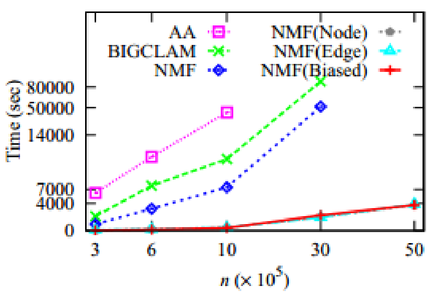
(d) YouTube



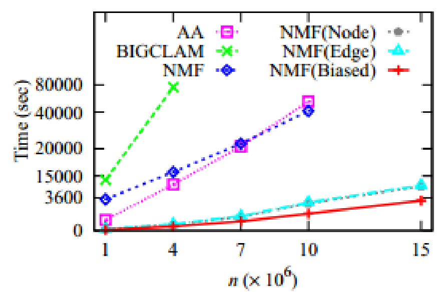
(e) Flickr



(f) Wikipedia



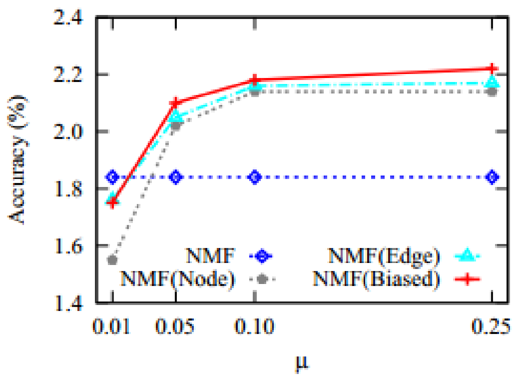
(g) Twitter



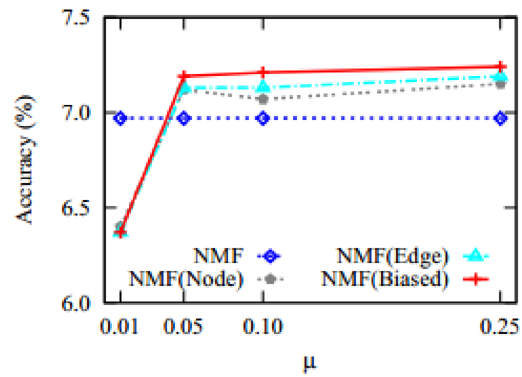
(h) Friendster

Experimental Results

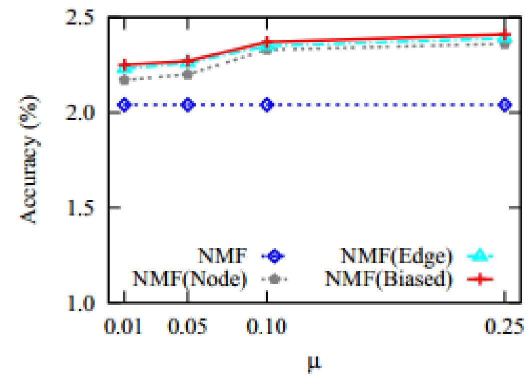
Accuracy and efficiency comparison: with respect to the expected appearing times μ



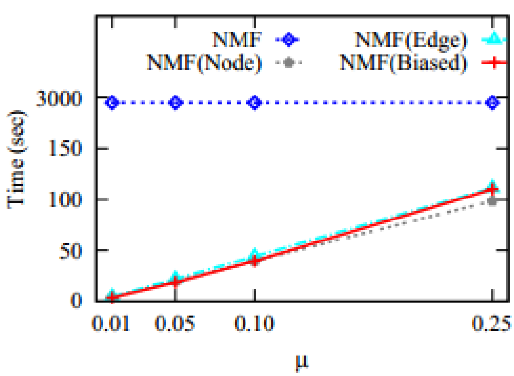
(a) YouTube



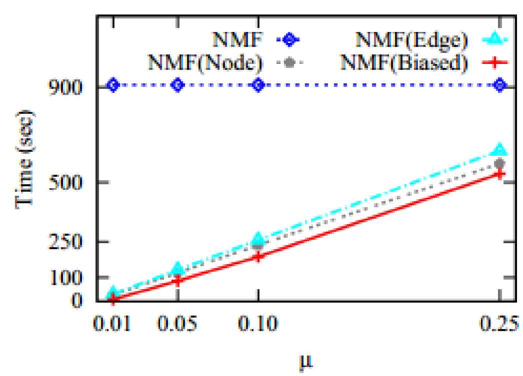
(b) Flickr



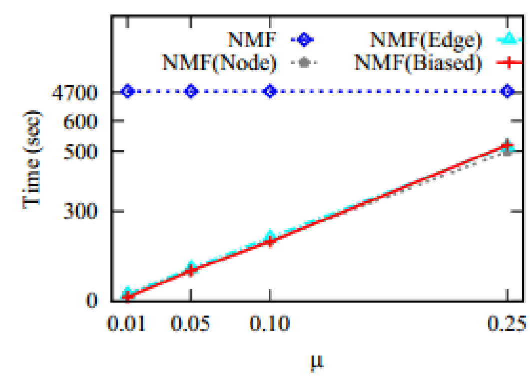
(c) Wikipedia



(d) YouTube



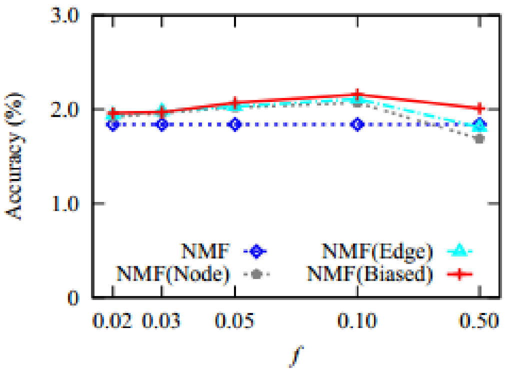
(e) Flickr



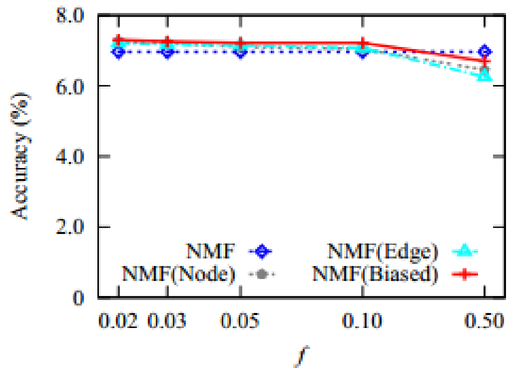
(f) Wikipedia

Experimental Results

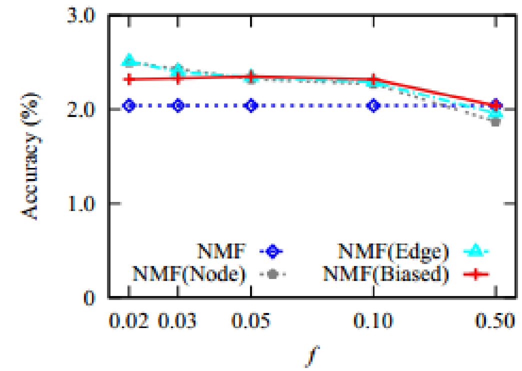
Accuracy and efficiency comparison: with respect to the fraction *f*



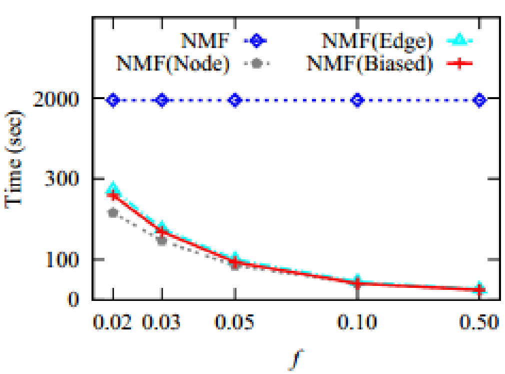
(a) YouTube



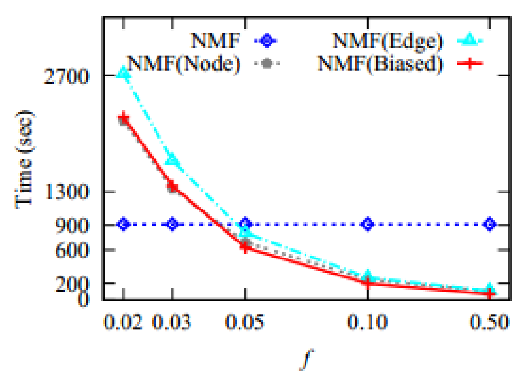
(b) Flickr



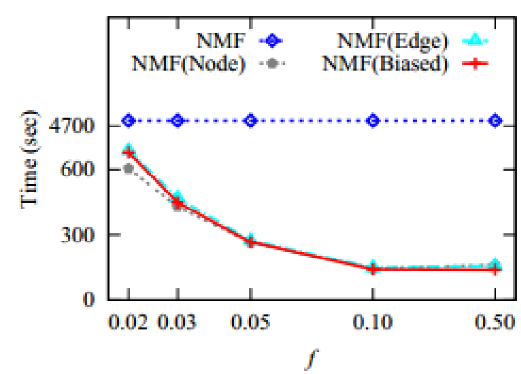
(c) Wikipedia



(d) YouTube



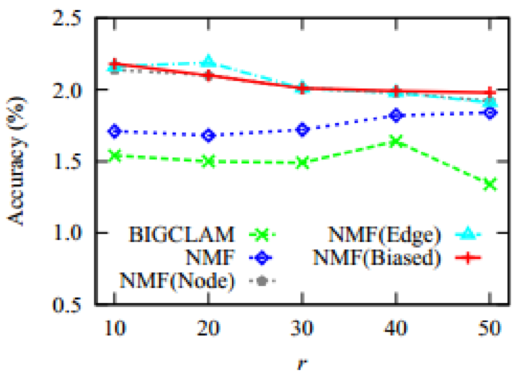
(e) Flickr



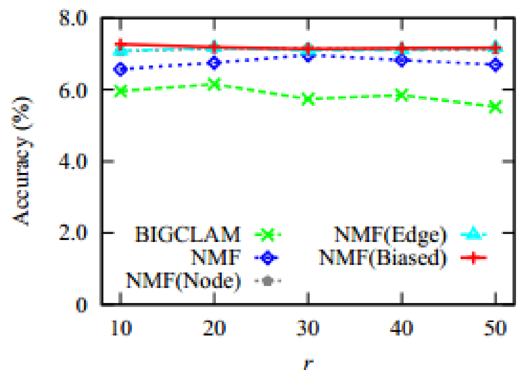
(f) Wikipedia

Experimental Results

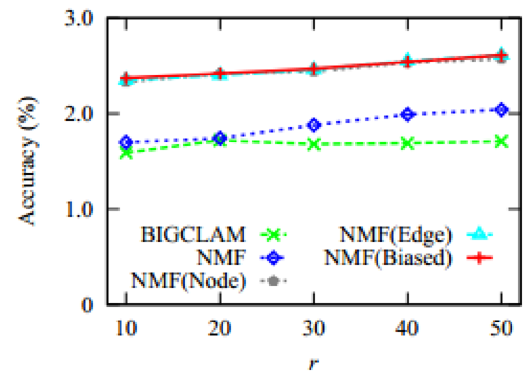
Accuracy and efficiency comparison: with respect to the number *r* of latent factors



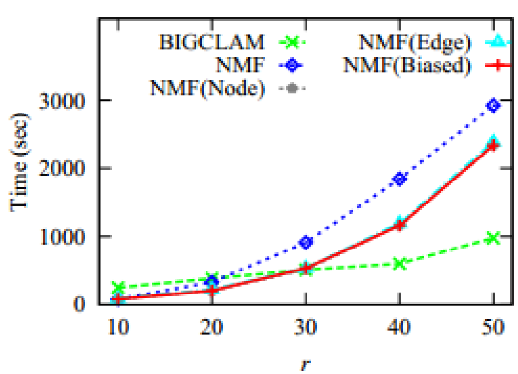
(a) YouTube



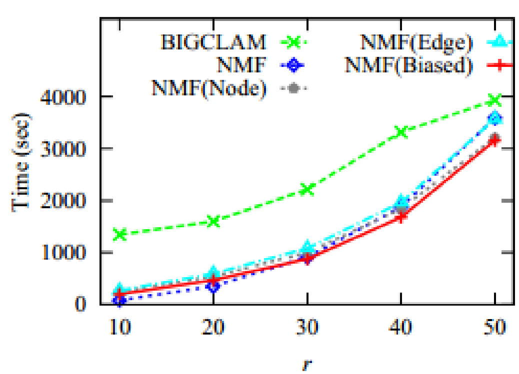
(b) Flickr



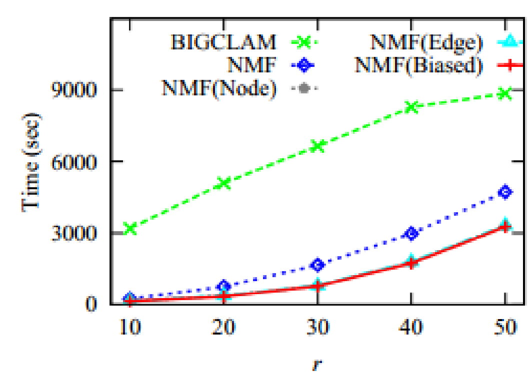
(c) Wikipedia



(d) YouTube



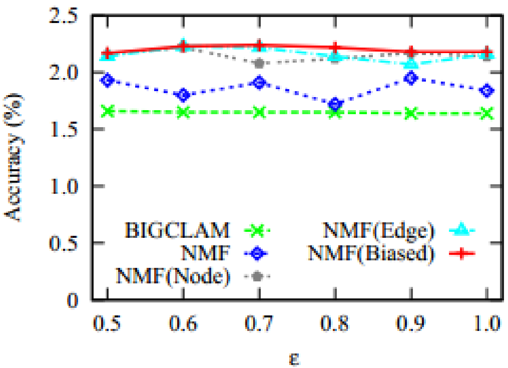
(e) Flickr



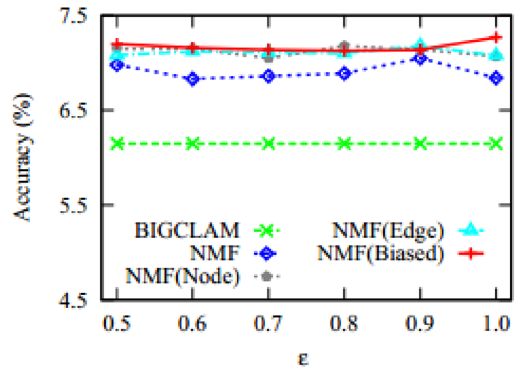
(f) Wikipedia

Experimental Results

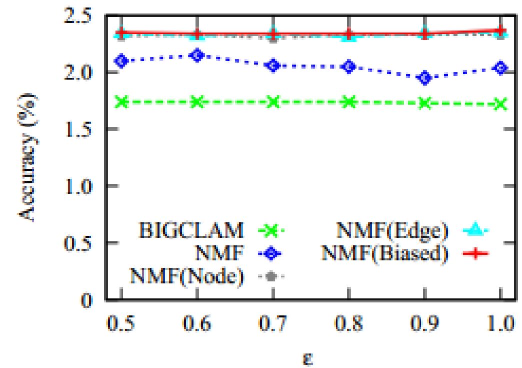
Accuracy and efficiency comparison: with respect to the tolerance ε of top-(ε, k)



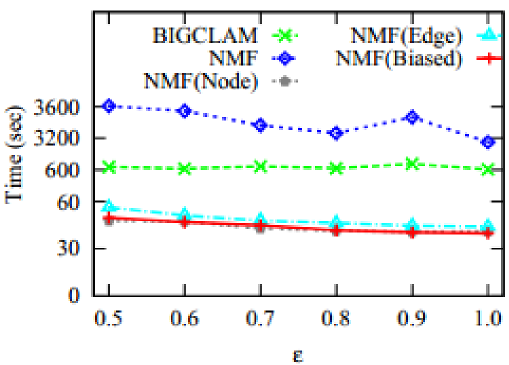
(a) YouTube



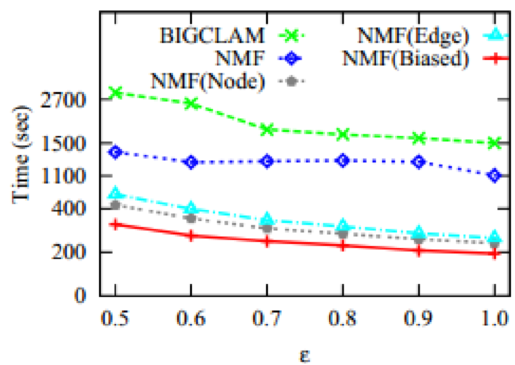
(b) Flickr



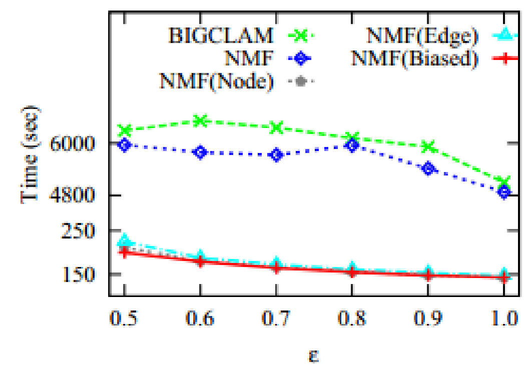
(c) Wikipedia



(d) YouTube



(e) Flickr



(f) Wikipedia