Scaling Pair-Wise Similarity-Based Algorithms in Tagging Spaces

Damir Vandic, Flavius Frasincar, Frederik Hogenboom

12th International Conference on Web Engineering
ICWE 2012

ERASMUS UNIVERSITEIT ROTTERDAM

Introduction

- Scalability often an issue when depending on pair-wise similarities (e.g., cosine similarity)
- Quadratic growth is a big problem
- Algorithms can not be applied to large data sets
  - heuristics used in most approaches

Our solution

- An algorithm that *approximately* filters *insignificant* (low) similarities
  - i.e., one only computes 'high' similarities
- We focus on tagging spaces (e.g., Flickr) and the cosine similarity
- Our approach can be applied to any similarity that depends on the dot product between two vectors

Overview
of the solution

Overview of the solution

tag	0	1	2	3	4	5
0	-	2	1	5	2	0
1	2	-	7	1	1	0
2	1	7	-	3	0	2
3	5	1	3	-	1	0
4	2	1	0	1	-	6
5	0	0	2	0	6	-

Overview of the solution

$(6 \times 6 - 6) / 2$
$=$
15 pairs

tag	0	1	2	3	4	5
0	-	2	1	5	2	0
1	2	-	7	1	1	0
2	1	7	-	3	0	2
3	5	1	3	-	1	0
4	2	1	0	1	-	6
5	0	0	2	0	6	-

Overview of the solution

(6 x 6 - 6) / 2
=
15 pairs

tag	0	1	2	3	4	5
0	-	2	1	5	2	0
1	2	-	7	1	1	0
2	1	7	-	3	0	2
3	5	1	3	-	1	0
4	2	1	0	1	-	6
5	0	0	2	0	6	-

Overview of the solution

tag	0	1	2
0	-	2	1
1	2	-	7
2	1	7	-
3	5	1	3
4	2	1	0
5	0	0	2

tag	3	4	5
0	5	2	0
1	1	1	0
2	3	0	2
3	-	1	0
4	1	-	6
5	0	6	-

Overview of the solution

tag	0	1	2
0	-	2	1
1	2	-	7
2	1	7	-
3	5	1	3
4	2	1	0
5	0	0	2

tag	3	4	5
0	5	2	0
1	1	1	0
2	3	0	2
3	-	1	0
4	1	-	6
5	0	6	-

The computed pairs are:

0-1	**3-4**
0-2	**3-5**
1-2	**4-5**

Overview of the solution

$2 \times (3 \times 3 - 3) / 2$
=
6 pairs

tag	0	1	2
0	-	2	1
1	2	-	7
2	1	7	-
3	5	1	3
4	2	1	0
5	0	0	2

tag	3	4	5
0	5	2	0
1	1	1	0
2	3	0	2
3	-	1	0
4	1	-	6
5	0	6	-

Algorithm (1)

Algorithm (I)

- How to choose the ‘dividing’ lines?
  - i.e., how to create the clusters of vectors?

Algorithm (1)

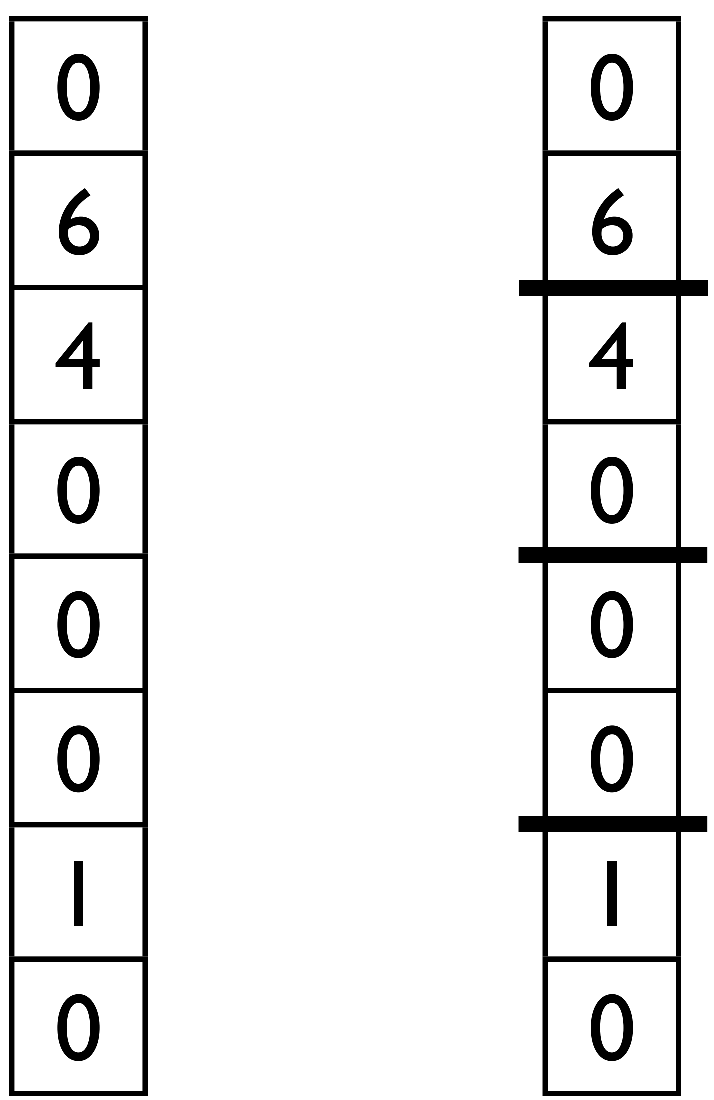
- How to choose the ‘dividing’ lines?
  - i.e., how to create the clusters of vectors?
- The algorithm:
  1. Compute for each vector (column) a hash
  2. Cluster all vectors that have the same hash

Algorithm (2)

0
6
4
0
0
0
1
0

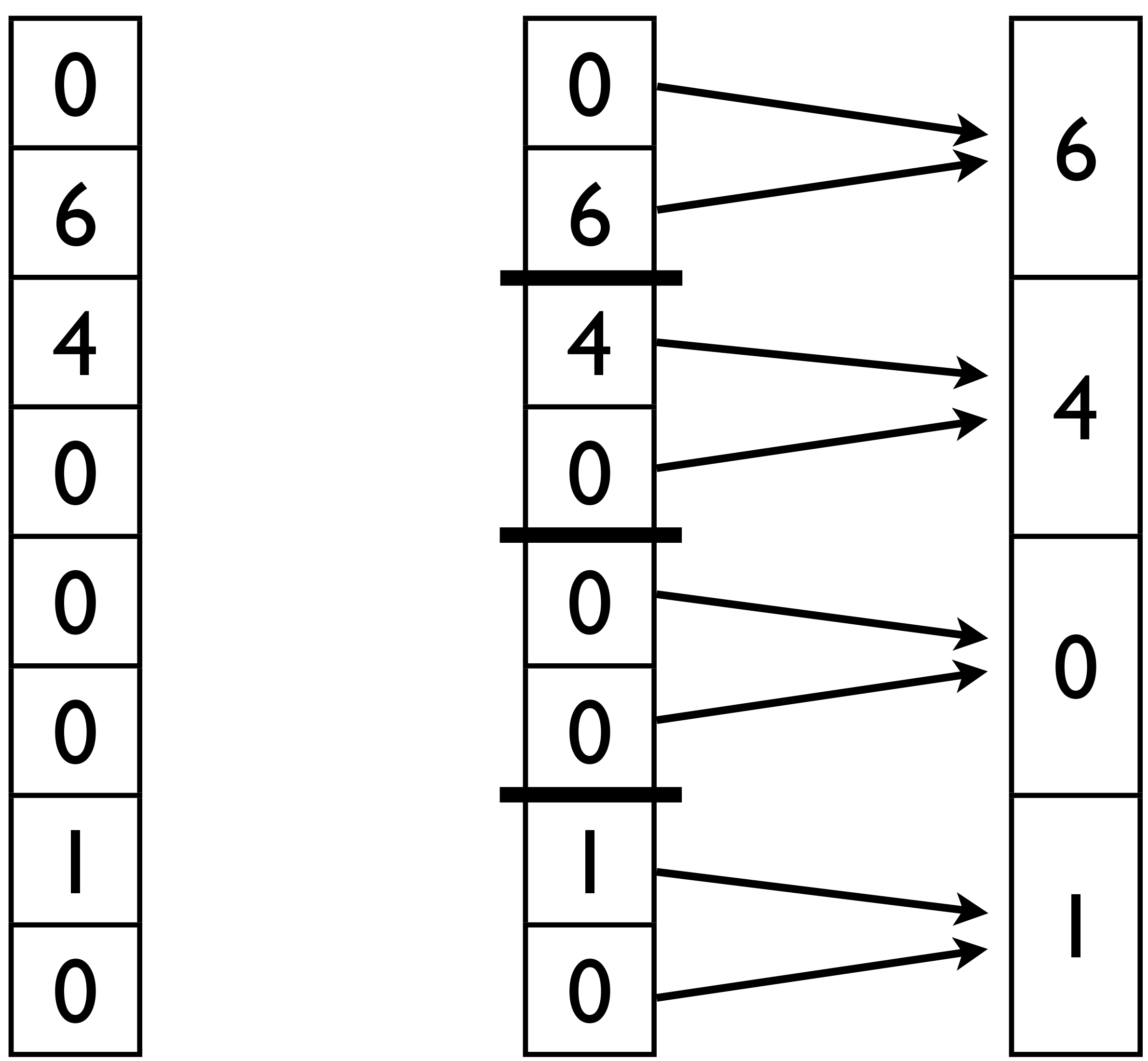
Algorithm (2)

Split in k parts

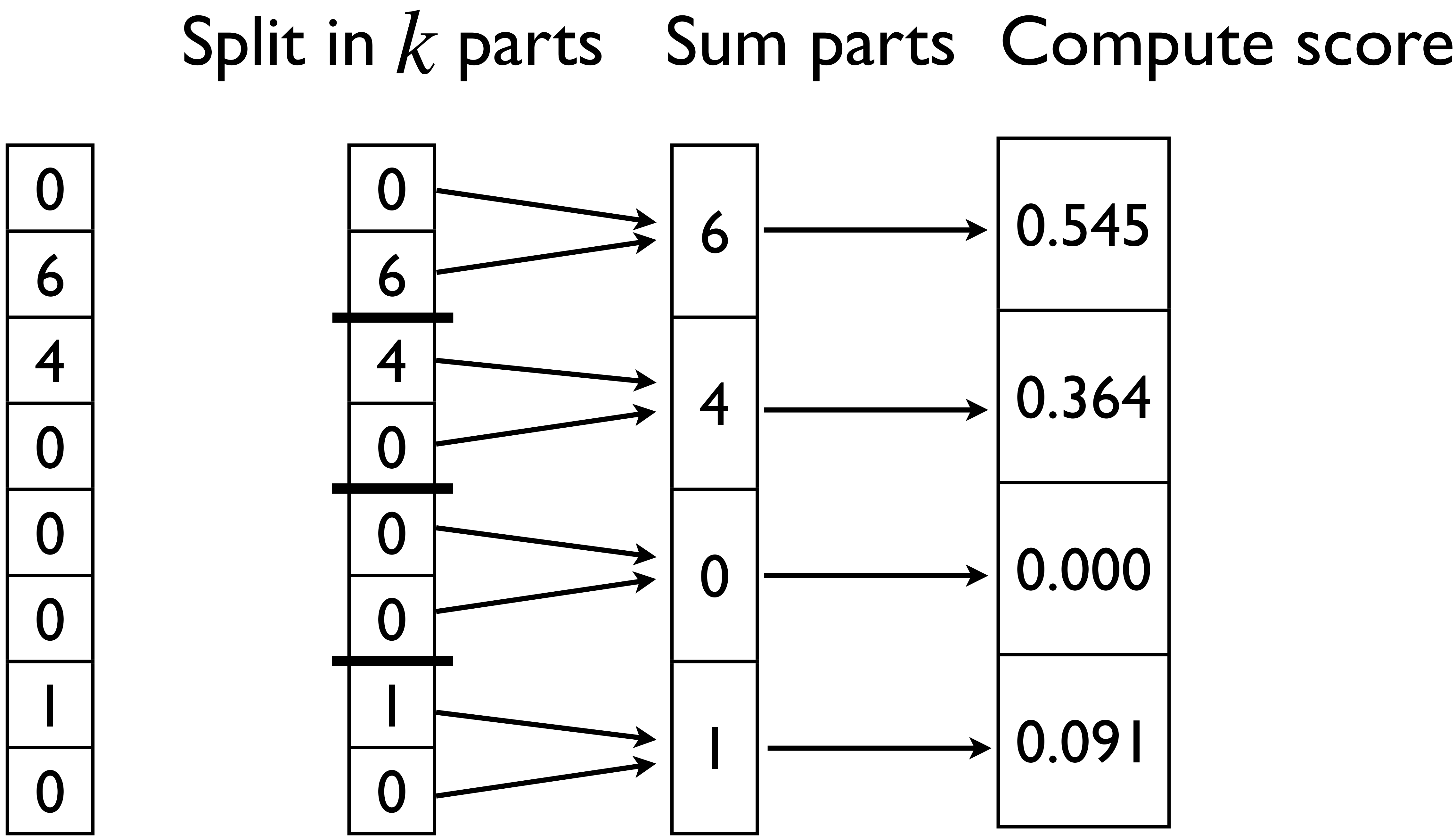


Algorithm (2)

Split in k parts Sum parts



Algorithm (2)


Split in k parts
Sum parts
Compute score
0
6
4
0
0
0
1
0
0
6
4
0
0
0
1
0
6
4
0
1
0.545
0.364
0.000
0.091

Algorithm (2)

Compute score

0.545
0.364
0.000
0.091

Algorithm (2)

Compute score　Compute hash (using α threshold)

0.545
0.364
0.000
0.091

Algorithm (2)

Compute score	Compute hash (using α threshold)
0.545	1
0.364	1
0.000	0
0.091	0

For $\alpha = 0.75$

Algorithm (3)

- Linear time complexity w.r.t. number of tags

$$O(n) = n(k \log k)$$

- For a given value for *k*, there are $2^k - 1$ possibilities for the hashes (i.e., clusters)
- The more clusters, the higher the reduction in the number of computations

Algorithm (4)

- Not only the number of clusters is important
  - How are the sizes of the clusters distributed?
  - When sizes are equal, the reduction is the largest

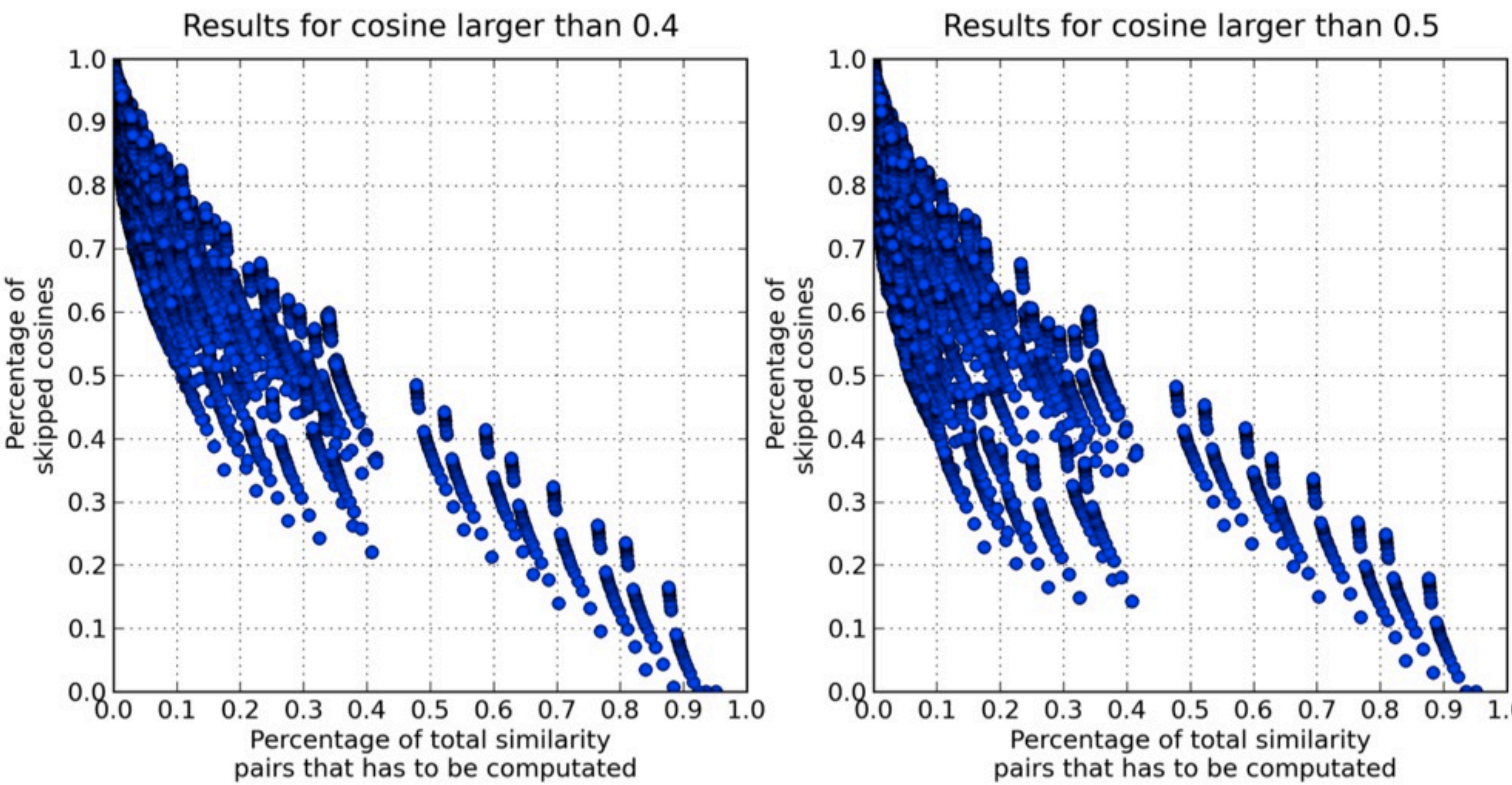
Evaluation

- We used a data set from Flickr
- Originally ~ 1.6 million tags
- We used top 50,000 occurring tags

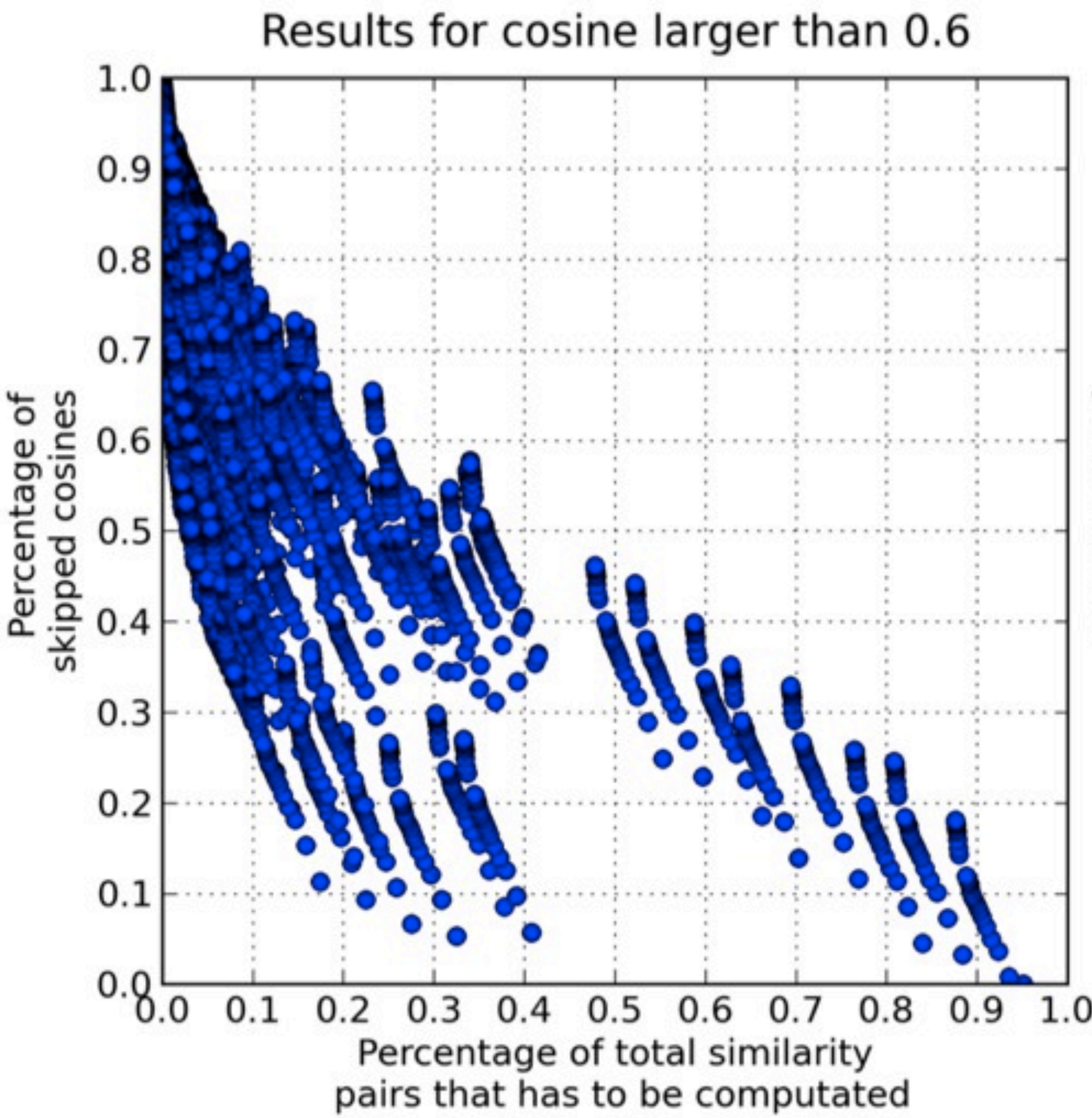
Evaluation

- Brute force evaluation of all cosines:
  - 1,249,975,000 cosines in total
- Run algorithm and record which cosines are skipped by the algorithm
- We performed our evaluation 30,720 times (for each unique parameter combination)
  - k -> from 3 to 50
  - α -> from 0.05 to 0.95 (step size: 0.05)

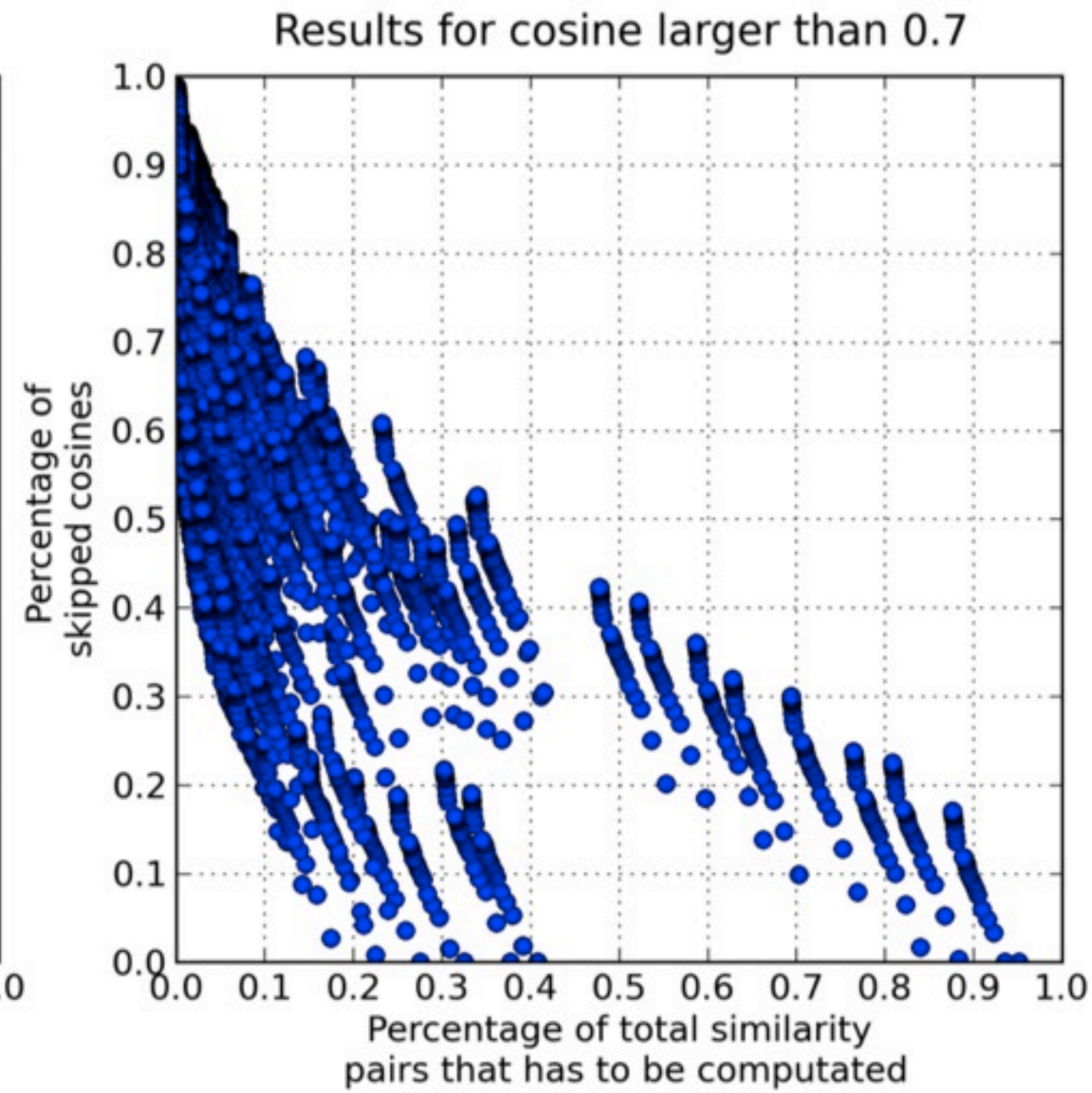
Evaluation


Results for cosine larger than 0.4
Percentage of skipped cosines
Percentage of total similarity pairs that has to be computed
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Results for cosine larger than 0.5
Percentage of skipped cosines
Percentage of total similarity pairs that has to be computed

Evaluation



Results for cosine larger than 0.6
Percentage of skipped cosines
Percentage of total similarity pairs that has to be computated
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0

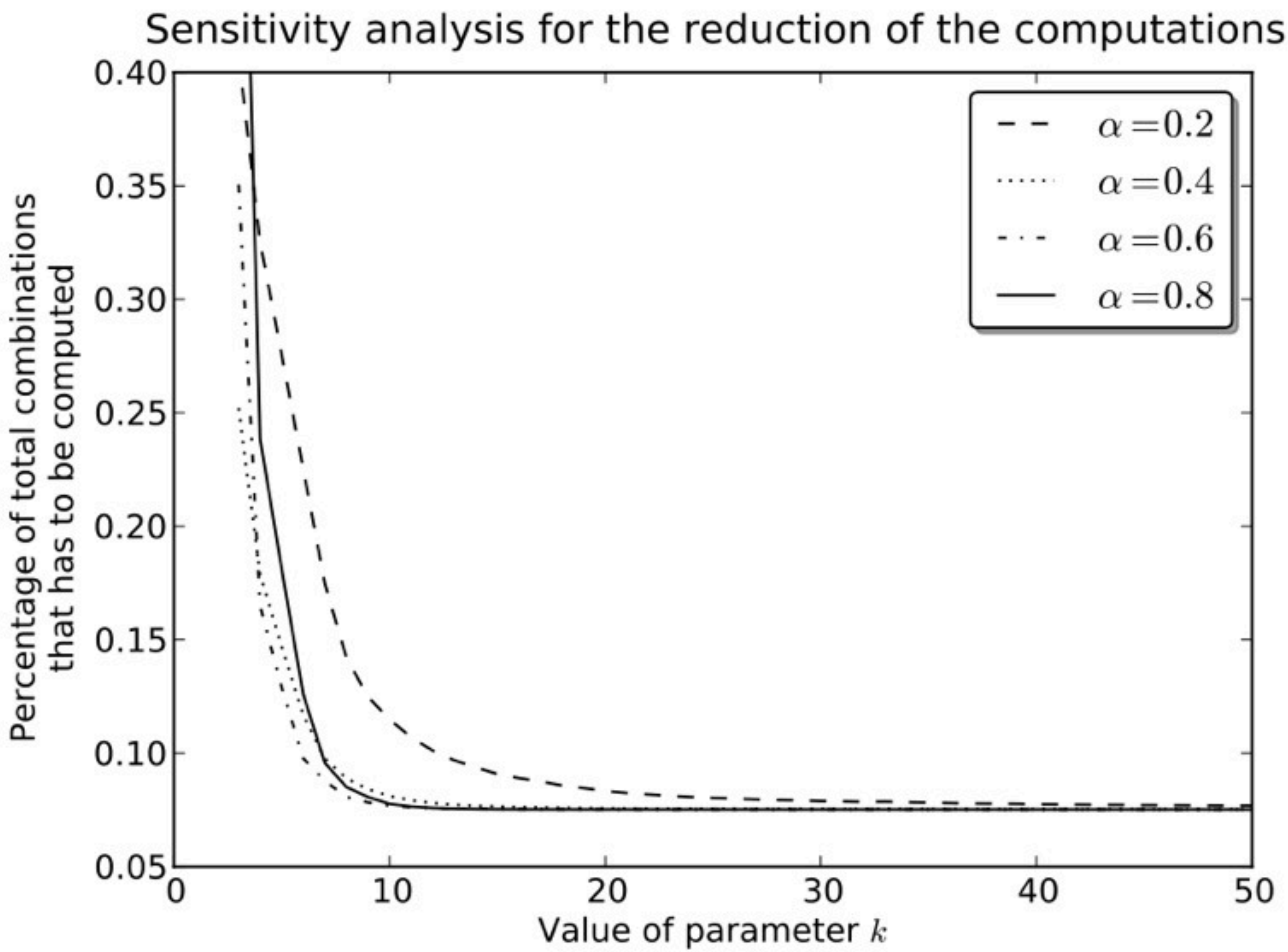


Results for cosine larger than 0.7
Percentage of skipped cosines
Percentage of total similarity pairs that has to be computated
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0

Evaluation

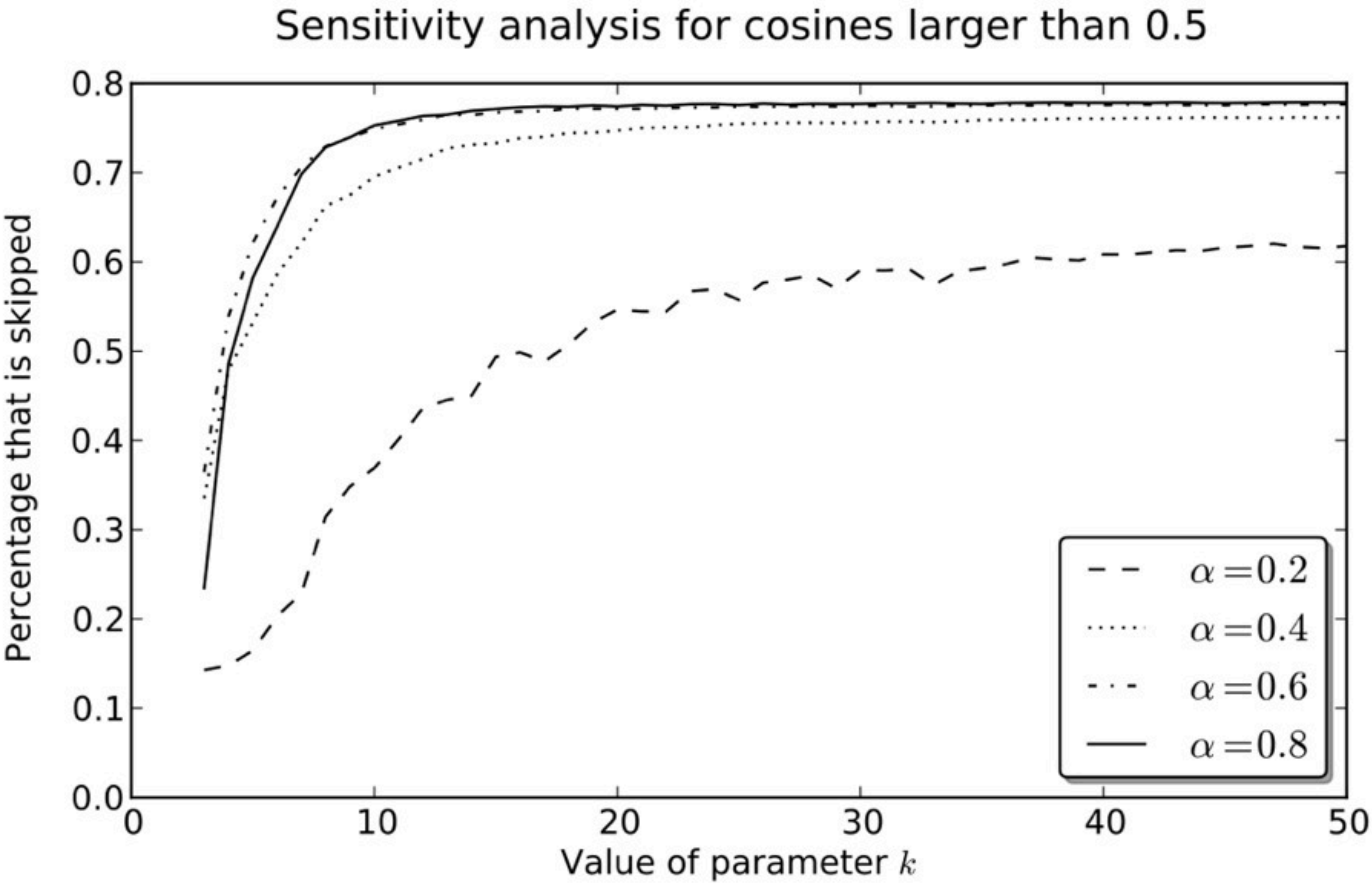
Threshold	Computations to be done	Skipped high cosines	Number of clusters	k	α
0.4	27.5%	27.1%	6	5	0.2
0.5	17.5%	22.9%	29	7	0.2
0.6	17.5%	11.3%	29	7	0.2
0.7	14.2%	8.8%	37	8	0.2
0.8	11.5%	5.6%	56	10	0.2
0.9	8.0%	1.0%	1309	14	0.3
0.4	76.9%	9.5%	7	3	0.85
0.5	40.8%	14.3%	4	3	0.3
0.6	22.5%	9.3%	6	5	0.2
0.7	17.5%	2.7%	29	7	0.2
0.8	17.5%	0.0%	29	7	0.2
0.9	8.2%	0.0%	2803	22	0.2

Evaluation

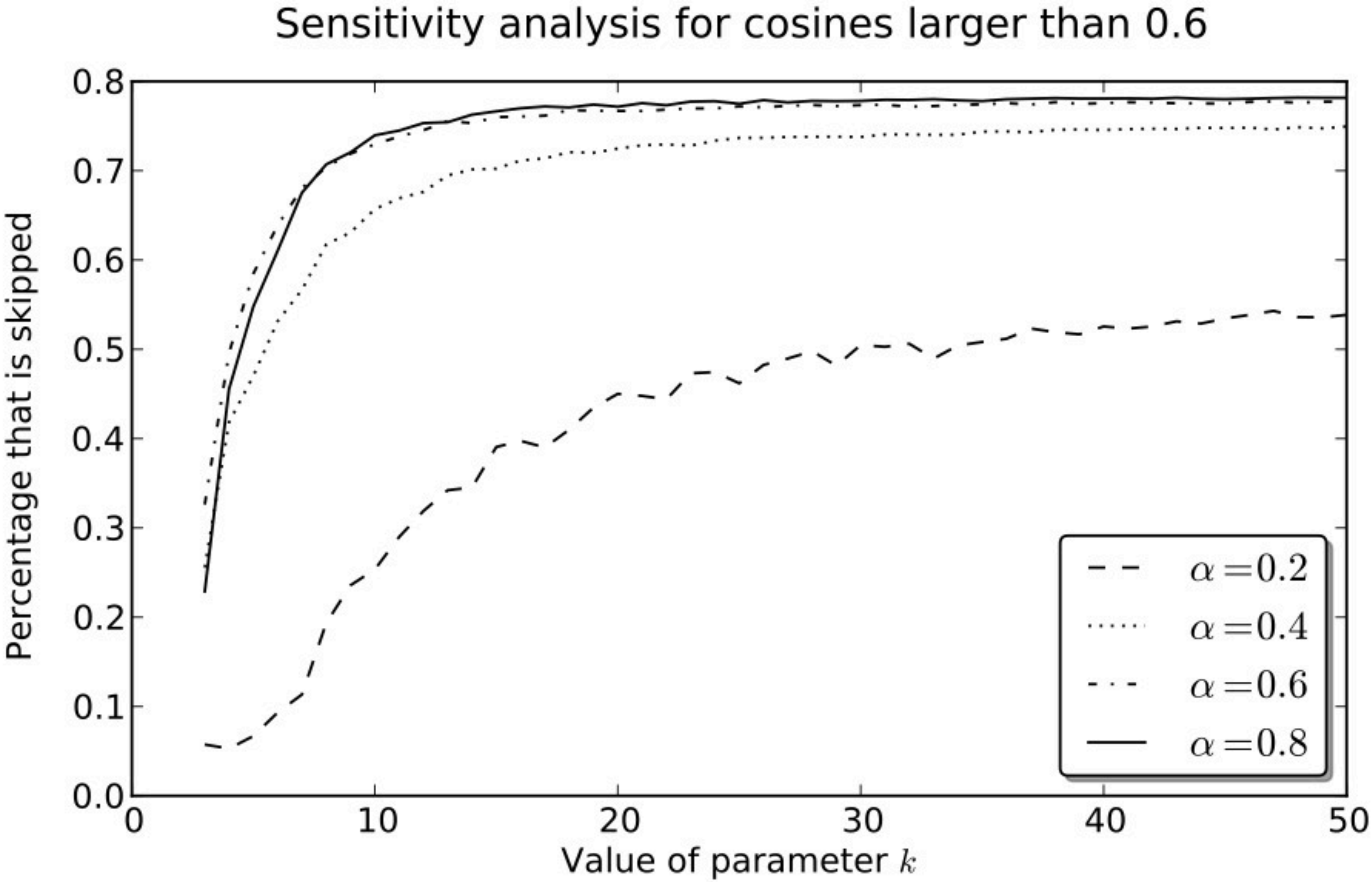


Sensitivity analysis for the reduction of the computations
Percentage of total combinations that has to be computed
Value of parameter k
$\alpha = 0.2$
$\alpha = 0.4$
$\alpha = 0.6$
$\alpha = 0.8$
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0
10
20
30
40
50

Evaluation


Sensitivity analysis for cosines larger than 0.5
Percentage that is skipped
Value of parameter k
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
10
20
30
40
50
$\alpha = 0.2$
$\alpha = 0.4$
$\alpha = 0.6$
$\alpha = 0.8$

Evaluation



Sensitivity analysis for cosines larger than 0.6
Percentage that is skipped
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
10
20
30
40
50
Value of parameter k
$\alpha = 0.2$
$\alpha = 0.4$
$\alpha = 0.6$
$\alpha = 0.8$

Conclusions

- Focused on the scalability issue that arises with the use of pair-wise similarities
  - Our approach uses binary hashes to cluster the vectors
  - The similarities are only computed within each cluster
- Results can be improved but are promising and useful in real-world applications

Questions?

Damir Vandic (vandic@ese.eur.nl)
http://damirvandic.com