EFFICIENT COMPRESSION ALGORITHM FOR MULTIMEDIA DATA

Karthik Revanuru

CRED

24th Sep 2020

High dimensional Multimedia data

- Text
  - bag-of-words representation
- Images
  - most pixels off when converting to black and white
  - Fourier spectrum of most real world images is sparse
- Interaction Matrices
  - user x item matrix in a recommendation system

Popular similarity/distance measures

- Inner Product
  - Number of common neighbors in social network
- Cosine Similarity
  - Text relevance
- Jaccard Similarity
  - User similarity in recommendation systems
- Euclidean Distance
  - Clustering
- Hamming Distance
  - Error correction

Similarity preserving dimensionality reductions

Why similarity preserving dimensionality reductions are useful ?

- Typically similarity sub-routines are called multiple times
- Compression $\rightarrow$ Efficient running time
- Compression $\rightarrow$ Efficient storage space
- Also serves as a regularization by pruning unimportant information

Our focus: sparse binary multimedia data

- Text
  - Bag of words
- Images
  - Black and white
- Interaction Matrices
  - User x item interaction

Our results

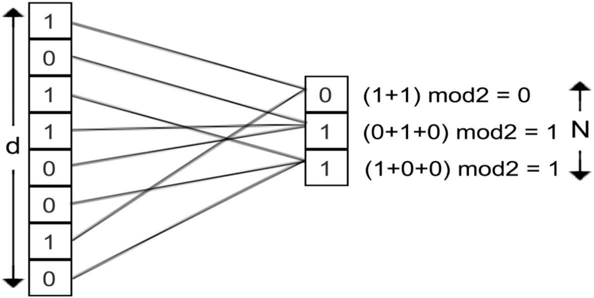
A simple and efficient dimensionality reduction for sparse binary data

- Binary to binary
- Earlier work preserves multiple similarity measures in one shot
  - Inner product
  - Jaccard similarity
  - Hamming distance
- In this work we show it preserves cosine similarity also
- Efficient
  - Fast
  - Space-efficient
  - Less randomness

Main Idea: Bucketing + XOR

- Partition the co-ordinates into k buckets randomly
- For each of the k bucket take XOR of the bits within it

Compression Scheme Diagram



Input vector V = (0, 1, 0, 1, 1, 0, 0, 0, 0, 0)
dim(V) = d = 10 and reduced dimension = N = 3
Random bucketing (b2, b1, b2, b2, b3, b1, b3, b1, b2, b3)
Output vector = (1, 1, 0)

Cosine Similarity

Consider a pair of binary vectors $\boldsymbol{u_i}, \boldsymbol{u_j} \in \{0,1\}^d$ such that the maximum number of 1s in any vector is at most ψ. If we set $N = O(\psi^2)$, and compress them into binary vectors $\boldsymbol{u_i^\backslash}, \boldsymbol{u_j^\backslash} \in \{0,1\}^N$ via BCS, then the following holds with high probability

$$Cos(u_i, u_j) = Cos(u_i^\backslash, u_j^\backslash)$$

Experimental Results - Dataset and Speedup

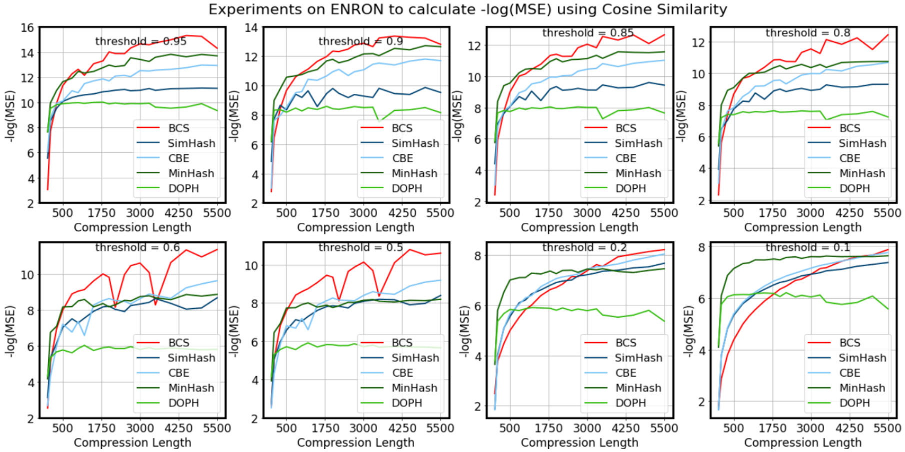
Two types of experiments

- MSE
- Ranking

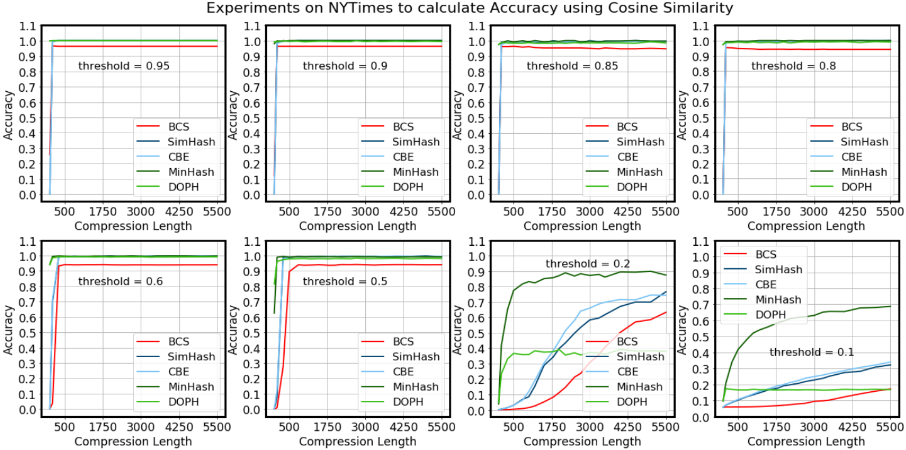
DataSet	Dimension	Speedup of BCS w.r.t SimHash	Speedup of BCS w.r.t. CBE	Speedup of BCS w.r.t MinHash	Speedup of BCS w.r.t DOPH
BBC	9635	108.66X	370.9X	130.2X	48.4X
Enron	28102	43.3X	233.6X	58.1X	48.01X
KOS	6906	50.8X	100.8X	69.2X	32.3X
NYTimes	102660	51.03X	158.16X	67.66X	56.87X

[1] Compressed Dimension is 500 in all cases.

Experimental Results - MSE Plot



Experimental Results - Ranking Plot



Experimental Results - Summary

- Improves running time by 100x ++
- Improves space storage by 32x ++
- Matches the benchmark accuracies
- Beats some of the benchmarks on downstream evaluations

Applications

- Recommendation systems
- Near-duplicate detection
- Hierarchical clustering
- Genome-wide association study
- Image & Audio similarity identification
- Digital video fingerprinting
- Extreme Classification

Thank You