Compressed Factorization:
Fast and Accurate Low-Rank Factorization of Compressively-Sensed Data

Vatsal Sharan*, Kai Sheng Tai*, Peter Bailis & Gregory Valiant
Stanford University

Poster 187

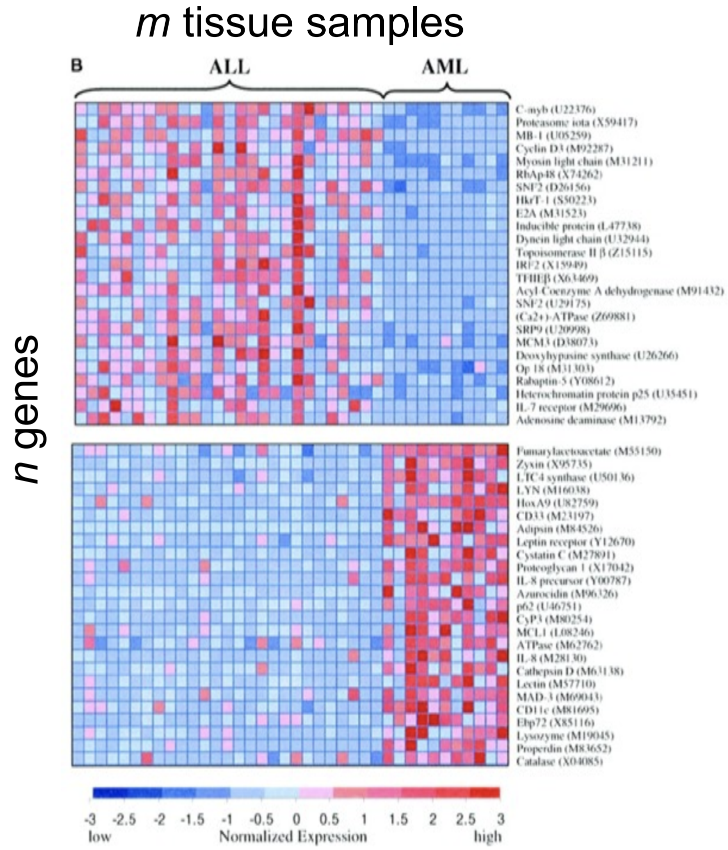
Learning from compressed data

- Suppose we are given data that has been compressed via random projections
  - e.g., in *compressive sensing* (Donoho'06, Candes+'08)
- What learning tasks can be performed directly on compressed data?
- Prior work:
  - support vector machines (Calderbank+'09)
  - linear discriminant analysis (Durrant+'10)
  - principal component analysis (Fowler'09, Zhou+'11, Ha+'15)
  - regression (Zhou+'09, Maillard+'09, Kaban'14)

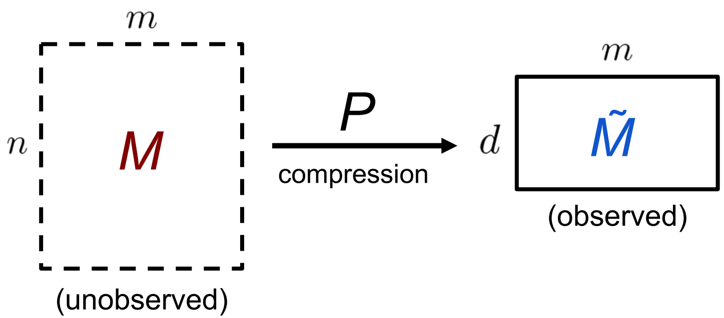
This work:
Low-rank matrix and tensor factorization of compressed data

Example: clustering gene expression levels

- Data: 2D matrix of gene expression levels
- Want to use nonnegative matrix factorization (NMF) to cluster data (Gao+'05)
- Compressive measurement (Parvaresh+'08)



Compressed matrix factorization: Setup



- Consider an n x m data matrix M with rank-r factorization $M = WH$, where W is a **sparse** matrix
- We observe **only** the compressed matrix $\tilde{M} = PM$ (the d x n *measurement matrix* P is known)
- **Goal:** recover factors W and H from the compressed measurements $\tilde{M}$

Two possible ways to do this

- Naïve way:
  - Recover the original data matrix using compressed sensing
  - Compute the factorization of this decompressed matrix

- Consider factorizing the data in compressed space:
  - Compute a sparse rank-r factorization $\tilde{M} = \tilde{W}\hat{H}$ (e.g., using NMF or Sparse PCA)
  - Run sparse recovery algorithm on each column of $\tilde{W}$ to obtain $\hat{W}$

- Computational benefit of factorizing in compressed space:
  - requires only $r \ll m$ calls to the sparse recovery algorithm
  - much cheaper than the naïve approach

Two possible ways to do this

- Naïve way:
  - Recover the original data matrix using compressed sensing
  - Compute the factorization of this decompressed matrix

- Consider factorizing the data in compressed space:
  - Compute a sparse rank-r factorization $\tilde{M} = \tilde{W}\hat{H}$ (e.g., using NMF or Sparse PCA)
  - Run sparse recovery algorithm on each column of $\tilde{W}$ to obtain $\hat{W}$

Its efficient, but does it even work?

When would compressed factorization work?

- Say we find the factorization $\tilde{M} = (PW)H$
- Then we can use sparse recovery to find W *from* (PW), as the columns of W are sparse.

$\tilde{M} = PM$
$M = WH$

Question: Since matrix factorizations are ***not*** unique in general, under what conditions is it possible to recover this “correct” factorization $\tilde{M} = (PW)H$, from which the original factors can be successfully recovered??

Our contribution

- Theoretical result showing that compressed factorization works under simple sparsity and low rank conditions on the original matrix.
- Experiments on synthetic and real data showing the practical applicability.
- Similar theoretical and experimental results for tensor decompositions.
- **Takeaway:**
  - Random projections can “preserve” certain solutions of non-convex, NP-hard problems like NMF
- See our poster for more details!

Poster #187

vsharan@stanford.edu
kst@cs.stanford.edu