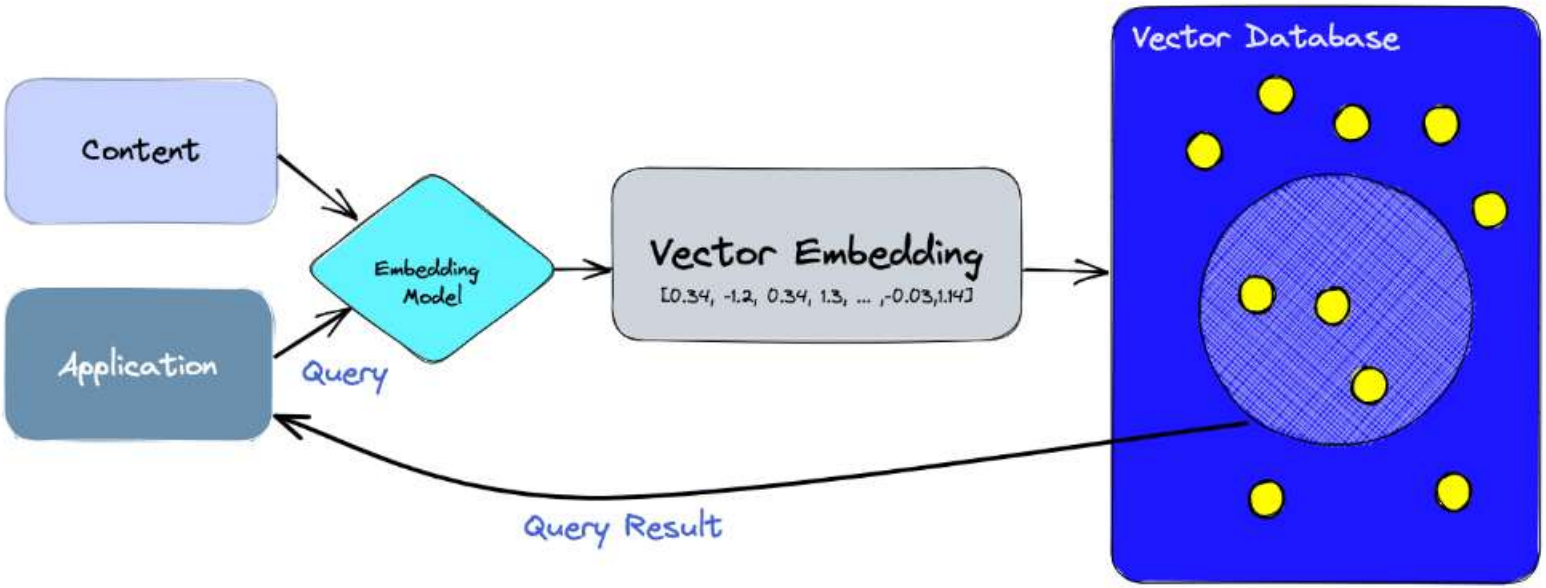
UVA | SCHOOL of DATA SCIENCE

Vector Stores and Random Projection

Distributed Computing
School of Data Science
University of Virginia

Vector Store Application: Semantic Search



Source: https://www.pinecone.io/learn/vector-database/

Vector Stores (Databases)

We learned that vector stores are purpose-built to manage vector embeddings

Semantic Search

We can do search via similarity matching (e.g., cosine similarity):

```
query = "which city has the highest population in the world?"

# create the query vector
xq = model.encode(query).tolist()

# now query
xc = index.query(xq, top_k=5, include_metadata=True)
```

```
[ ]  for result in xc['matches']:
         print(f"{round(result['score'], 2)}: {result['metadata']['text']}")
```

```
0.88:  Which is the most populated city in the world.?
0.88:  What is the most populated city in the world?
0.81:  Which is the largest city in the world?
0.8:  Which is biggest city in the world?
0.79:  What's the world's largest city?
```

Approximate Nearest Neighbor (ANN) Search

Searching through vector store is expensive at scale

ANN search uses algorithms to more quickly return a sufficient match

Trades off between accuracy and speed

A good system can be fast and highly accurate

One important algorithm is ***random projection***

Random Projection: Overview

Project high-dimensional (*d*) vectors into lower-dimensional subspace *k*

Uses **random projection matrix**

What are the important attributes?

Random Projection in Dimensionality Reduction: Applications to Image and Text Data.
E. Bingham and H. Mannila.

Random Projection: Overview

Project high-dimensional (d) vectors into lower-dimensional subspace k

Uses **random projection matrix**

What are the important attributes?

- columns have unit lengths
- $k << d$
- needs to preserve similarity measures

Random Projection in Dimensionality Reduction: Applications to Image and Text Data.
E. Bingham and H. Mannila.

Random Projection: Findings

In theory, the method preserves similarity measures
In practice, not many published results

Paper shows good results for
- noisy and noiseless images
- information retrieval from text documents

Similarity is preserved AND method is faster than competitors like PCA

Random Projection: How it Works

In random projection, the original d-dimensional data is projected to a k-dimensional ($k << d$) subspace through the origin, using a random $k \times d$ matrix R whose columns have unit lengths. Using matrix notation where $X_{d \times N}$ is the original set of N d-dimensional observations,

$$X^{RP}_{k \times N} = R_{k \times d} X_{d \times N} \quad (1)$$

Random Projection: Why does it Work?

Johnson-Lindenstrauss lemma

If points in vector space are projected onto randomly selected subspace of *suitably high dimension*, distances between points are approximately preserved

Research Questions and Methods

How much is similarity of two vectors distorted by RP?
How efficient is this method?

Similarity (e.g., Euclidean distance, dot product)

Compare various methods of dim reduction for range of *k*
For each k, compute new R and projection matrix

Research Questions and Methods

How much is similarity of two vectors distorted by RP?
How efficient is this method?

Similarity (e.g., Euclidean distance, dot product)

Compare various methods of dim reduction for range of *k*
For each k, compute new R and projection matrix

Methods:
- Random Projection (RP)
- Sparse Random Projection (SRP)
- Discrete Cosine Transform (DCT)
- Singular Value Decomposition (SVD)

RP and SRP

RP generally samples from Gaussian distn

SRP can use a simpler distribution such as:

$$r_{ij} = \sqrt{3} \cdot \begin{cases} +1 & \text{with probability } \frac{1}{6} \\ 0 & \text{with probability } \frac{2}{3} \\ -1 & \text{with probability } \frac{1}{6}. \end{cases}$$

RP and SRP

RP generally samples from Gaussian distn

SRP can use a simpler distribution such as:

$$r_{ij} = \sqrt{3} \cdot \begin{cases} +1 & \text{with probability } \frac{1}{6} \\ 0 & \text{with probability } \frac{2}{3} \\ -1 & \text{with probability } \frac{1}{6}. \end{cases}$$

What is $E[r_{ij}]$

Experiment 1: Noiseless Image Data

N = 1000 image windows drawn from 13 monochrome images of natural scenes

Cropped 256x256 pixel images down to 50x50 windows

Each window presented as *d*-dim column vector (*d* = 2500)

Experiment 1: Noiseless Image Data

N = 1000 image windows drawn from 13 monochrome images of natural scenes

Cropped 256x256 pixel images down to 50x50 windows

Each window presented as d-dim column vector (d = 2500)

1 | Compute Euclidean distances between pairs of original vectors.
2 | Compute Euclidean distances between pairs of dim reduced vectors.
3 | Compute the diff.
4 | Average over 100 pairs. Call this the *error*.

Experiment 1: Noiseless Image Data; Errors

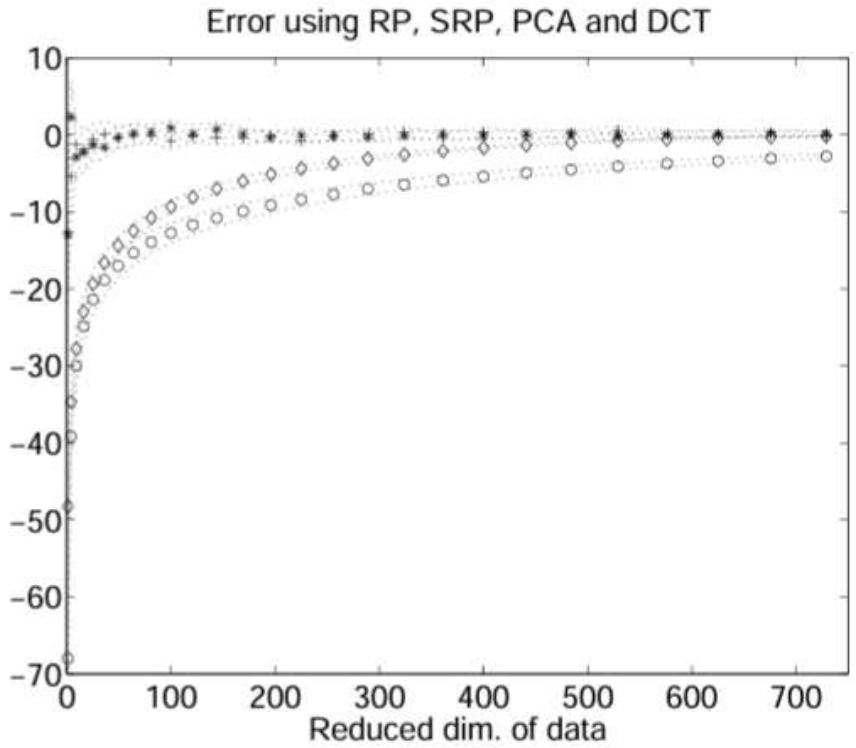


Figure 1: The error produced by RP (+), SRP (∗), PCA (⋄) and DCT (∘) on image data, and 95 % confidence intervals over 100 pairs of data vectors.

Experiment 2: Text Data

N = 2262 documents drawn from 20 newsgroups corpus
Docs converted to term frequency vectors
Removed common terms
Normalized document vectors to unit length
Vocab size: 5000

Experiment 2: Text Data

N = 2262 documents drawn from 20 newsgroups corpus
Docs converted to term frequency vectors
Removed common terms
Normalized document vectors to unit length
Vocab size: 5000

1 | Compute dot product between pairs of original vectors.
2 | Compute dot product between pairs of dim reduced vectors.
3 | Compute the diff.
4 | Average over 100 pairs. Call this the *error*.

Experiment 2: Text Data; Errors

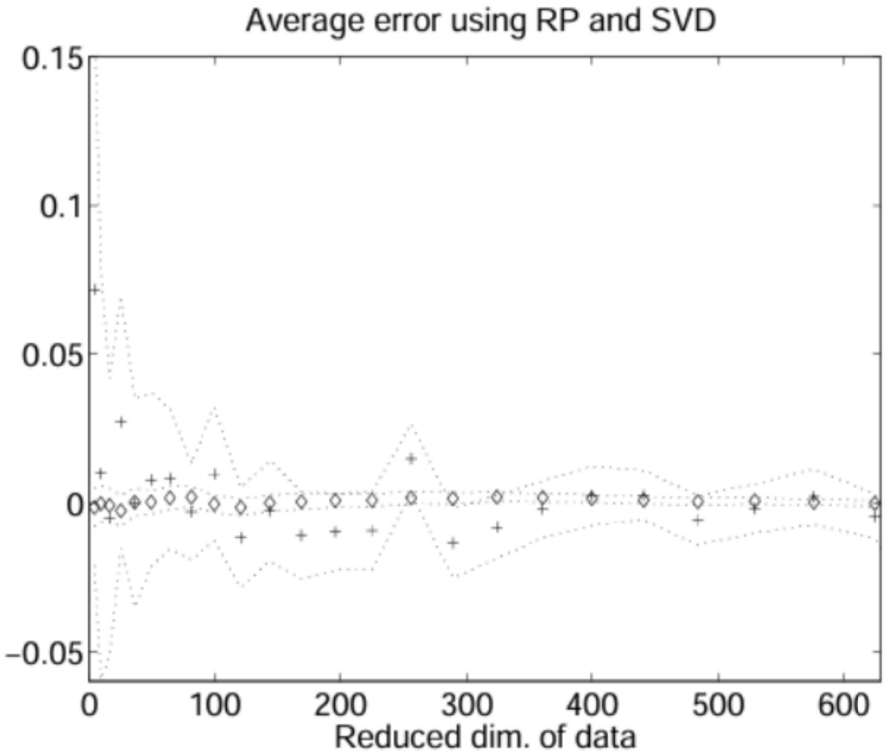


Figure 4: The error produced by RP (+) and SVD (⋄) on text document data, with 95% confidence intervals over 100 pairs of document vectors.

SVD works better but is much more computationally expensive

Conclusions

Evidence that RP preserves similarity of data vectors and it's fast

For images, RP worked better than PCA, DCT

For text, RP worked well but not as well as SVD

When there is high dimensional data, RP can be a good alternative to computationally infeasible methods

Thought Experiment

Given the semantic search lab, how could we apply RP to the data?