Wilcoxon Signed-Rank Test to Compare Document Embedding Algorithms

Anton Chen
contact@antonchen.ca
April 5, 2024

STAT 461 Statistical Inference II
The University of British Columbia
Vancouver, B.C.

Contents

1 Introduction

1.1 Motivation: what does this have to do with STAT 461?

Even for the most pure of mathematicians in this class, it's a good feeling to know that what we're learning has use in the real world. The central aim of this project is to demonstrate how to approach a less-than-well-defined problem and set up sufficient context such that the methodology we've learned can be applied properly. To illustrate this, we'll walk through an application: the task of dimensionality reduction, and more specifically, comparing document embedding algorithms.

1.2 Note

I was debating between this topic or a proof walkthrough of $X_n \xrightarrow{a.s.} X \implies X_n \xrightarrow{p} X$, but I think this is much more interesting. This is more in line with my passion. I hope you enjoy.

2 Context

We have to understand our problem before we try to solve it. Let's do that now.

2.1 High-dimensional data

Definition 2.1 (High-Dimensional Dataset)**.**

$$\mathbf{X} \in \mathbb{R}^{n \times d} \text{ where } d \text{ is large.}$$

As you would expect, high-dimensional data is data with many dimensions. In application, it's becoming increasingly more relevant, common, and important. Applications are widespread, including in finance, genomics, and so on [1].

As dimensionality increases, we face a few problems. The main one is computational cost — many datasets are simply intractible to work with if left in high-dimension [4]. In addition, high-dimensional datasets in practice tend to have redundant or irrelevant features [3]. Let's also not forget about the curse of dimensionality either [8].

2.2 Dimensionality reduction (DR)

The common antidote is to reduce the dimensionality of the data, a.k.a. dimensionality reduction (or DR). DR is implemented with a DR algorithm.

Definition 2.2 (DR Algorithm)**.**

$$g = \left\{ g_{d_{\text{reduced}}} : \mathbb{R}^{n \times d} \to \mathbb{R}^{n \times d_{\text{reduced}}} \text{ where } 0 < d_{\text{reduced}} \leq d \right\}$$

Interpretation: a group of mappings from the original feature space to lower dimension feature space, where we can choose the target dimension. In practice, a desirable quality is that it retains some semblance of "quality" or "meaning" in the data after reduction. It turns out that this is a rather vague concept, so there's many ways to measure this.

2.2.1 Example: PCA

> Principal Component Analysis (PCA) is probably the most famous DR algorithm, and you probably know it already. As a reminder, it projects points onto the subspace spanned by the d_{reduced} principal components, where d_{reduced} is our target dimensionality.

There are many DR algorithms, such as LDA, Isomap, t-SNE, etc. The key point is that we have many DR algorithms at our disposal, and we'd like to know which ones perform the best for our situation.

2.3 Quality measures

Definition 2.3 (Quality Measure)**.** A quality measure q is a function

$$q\colon \mathbb{R}^{n\times d} \times \mathbb{R}^{n\times d'} \to [0,1]$$

A quality measure is simply a way to score how good of a job a DR algorithm performs on a dataset [7]. It's just a function. The mapping $q = 0$ is a quality measure, albeit rather useless. The convention is that a score closer to 1 indicates a better quality reduction.

Key examples include explained variance ratio, reconstruction error, silhouette score. However, depending on our goals, we may need more sophisticated quality measures.

2.4 Problem statement

In the context of DR, if we've identified optimality criterion (i.e. a quality measure), have some DR algorithms in mind, and have a dataset, a natural question arises. Which algorithm is the best?

This leads us to the big question of this project:

> Given dataset $\mathbf{X} \in \mathbb{R}^{n\times d}$, DR algorithms g^1, g^2, and quality measure q, determine whether g^2 yields higher quality reductions than g^1.

It turns out we can now very easily tackle this problem with methodology from class.

3 Review

If you've noticed already, the problem statement seems reminiscent of the Wilcoxon signed-rank test for paired samples. We'll give a brief reminder of what that entails.

3.1 Wilcoxon signed-rank test for paired samples

Suppose we have n paired samples (x_i, y_i). For the sake of simplicity, let's filter these into s.t. $x_i \neq y_i$ for $i = 1, \cdots, n'$. In particular, $X_i \sim F$ and $Y_i \sim G$. Our hypotheses are $H_0\colon F \equiv G$ v.s. $H_1\colon F < G$. One important note to make about F, G are the lack of assumptions we make about them. We do not assume they are regular nor parametric. This poses importance in applications where we simply cannot make such assumptions about F, G.

Let's review some notations. Let $\Delta_i := |y_i - x_i|$ and $\delta_i := \begin{cases} 1 & y_i > x_i \\ -1 & \text{o.w.} \end{cases}$ represent the magnitude and sign of the paired sample differences. In addition, let

$$R_i := \sum_{j=1}^{n} \mathbb{1}\left\{\Delta_j < \Delta_i\right\} + \frac{1}{2}\sum_{j=1}^{n} \mathbb{1}\left\{\Delta_j = \Delta_i\right\} + \frac{1}{2}.$$

As Prof. Chen mentioned in his notes, this scary looking formula has the interpretation of sample i's rank of the difference magnitude [2]. We arrive at the test statistic and test.

Definition 3.1 (Wilcoxon Signed-Rank Test Statistic)**.**

$$W_n = \sum_{i=1}^{n} \delta_i R_i.$$

Definition 3.2 (Wilcoxon Signed-Rank Test)**.**

$$\phi(W_n) = \mathbb{1}\left\{W_n > z_{1-\alpha}\sqrt{\frac{1}{6}n(n+1)(2n+1)}\right\}. \qquad \text{(For a specified test size of } \alpha\text{)}$$

$z_{1-\alpha}$ denotes the $(1-\alpha)$th quantile of $\mathcal{N}(0,1)$. A key observation is that W_n is asymptotically normal, justifying the square root term and $z_{1-\alpha}$. Alternatively, p-value $p = \mathbb{P}[W_n > w_0]$ can be used to determine whether to reject H_0 or not.

That basically wraps up the review. Nothing should be new here. Let's go back to tackling our problem.

4 Methodology

From here, we'll further refine our problem statement into context appropriate to apply Wilcoxon's signed-rank test. The methodology outline here closely follows Gracia et al. [5]

4.1 Select range of target dimensions D

We want to compare g^1 and g^2 across various target dimensions of interest. Denote

$$D = \{d_{\text{reduced}} \colon d_{\text{reduced}} \text{ is of interest}\}. \qquad \text{(w.r.t. our problem)}$$

In practice $D = \{2, 3, \cdots, d\}$ is used [5], but we could also add a stride to D, e.g. $D = \{d_{\text{reduced}} = 2, 3, \cdots, d \colon d_{\text{reduced}} \mod 5 = 0\}$.

4.2 Extract quality scores from dataset

We want to form pairwise quality scores corresponding to each target dimension. Mathematically speaking, we compute

$$\mathbf{Q}_{ij} = q(\mathbf{X}, g^j_{d_i}(\mathbf{X})) \qquad \text{(For } d_i \in D,\ j = 1, 2)$$

where $g^j_{d_i}$ is the reduced dataset to d_i dimensions via DR algorithm g^j, and q is our quality measure. This yields pairwise samples for which we use to compare the quality of the two algorithms.

4.3 Apply Wilcoxon signed-rank test on quality scores

We've finally made it. Simply apply the test on the paired samples! Think for a moment about how we interpret the result of the test w.r.t. our context.

5 Example

Here's the fun part. Let's apply this methodology to a real problem and dataset.

5.1 Application: document embedding for NLP

Documents (a sequence of words) are commonplace in natural language processing and have the unfortunate characteristic of being difficult to represent mathematically and computationally. Think about it. How you would represent a document? How would you represent 10 billion documents?

One common way to represent a document is with a **Bag of Words** (BoW) approach [10]: simply count the frequencies of each word in the document.

The dimension d ends up being the size of your vocabulary. This doesn't see very good — think of how many words are in the English language! Think about other languages! Clearly modern NLP approach do not utilize such document representations. Perhaps we can use DR to mitigate such extravagant dimensionality.

Notable modern approaches include **Doc2Vec**, originating from Word2Vec, which learn document representation via skip-grams (think of sliding windows about each word). Another approach is **Latent Semantic Analysis (LSA)**, which applies SVD on the term-document matrix. Other notable approaches definitely exist, but we'll focus on these 2 for now.

Table 1: Term-document matrix $\mathbf{X} \in (\mathbb{Z}_+ \cup \{0\})^{n \times d}$

Doc/Term	**the**	**quick**	**brown**	**fox**	$\cdots$
Doc 1	1	1	1	1	$\cdots$
Doc 2	0	1	0	6	$\cdots$
Doc 3	0	100	1	6	$\cdots$
Doc 4	0	0	0	1	$\cdots$
Doc 5	0	0	0	0	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

Another tricky topic is quality measures. Quality measures are often defined w.r.t. a downstream task, e.g. classification, sentiment analysis, and so on. Different document representations seem to be more optimal for different tasks. In general, some qualities of a good quality measure include: non-conflation, robustness against lexical ambiguity, demonstration of multifacetedness, reliability, etc. [9]

Let's proceed with the experiment.

5.2 Experiment design

The `20newsgroups` dataset is a collection of approximately 18 000 newsgroups posts spanning 20 different categories [6]. A task we may be interested in is **document classification**: given the contents of a newsgroups post, can we identify its topic? As mentioned prior, it is wise to perform DR on the dataset. Denote g^1 as our LSA-based DR algorithm, and g^2 as our Doc2Vec-based DR algorithm. Since our task is classification, our quality measure may indicate how well DR supports classification. In this experiment we use training accuracy of logistic regression with reduced datasets as training data. Indeed this is rather contrived, and better measures certainly exist out there, but this is moreso for illustrative purposes. For now, define quality measure

$$q(X_{\text{reduced}}) = \text{training classification accuracy from training logistic regression on } X_{\text{reduced}}.$$

On to the implementation.

5.3 Implementation

Before we start, let's import everything we'll be using.

```
import numpy as np
from scipy.stats import wilcoxon
from sklearn.datasets import fetch_20newsgroups
from sklearn.decomposition import TruncatedSVD
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from gensim.models.doc2vec import Doc2Vec, TaggedDocument
```

Once that's done, let's implement our DR algorithms. Firstly, Doc2Vec:

```
def reduce_doc2vec(X, output_dim=50):
  tagged_data = [
      TaggedDocument(words=d.split(), tags=[str(i)]) for i, d in enumerate(X)
  ]
```

```
  model = Doc2Vec(
      tagged_data,
      vector_size=output_dim,
      window=5,
      min_count=5,
      epochs=3,
  )

  X_doc2vec = np.array([model.infer_vector(doc.split()) for doc in X])

  return X_doc2vec
```

Next, LSA:

```
def reduce_lsa(X, output_dim=50):
  # Convert text documents to a document-term matrix
  vectorizer = CountVectorizer()
  X_counts = vectorizer.fit_transform(X)

  # Apply SVD
  lsa = TruncatedSVD(n_components=output_dim)
  X_lsa = lsa.fit_transform(X_counts)

  return X_lsa
```

Following the methodology, let's compute the quality scores. Depending on your hardware, this may take a while.

```
def get_quality_scores(X, y, output_dims):
  quality_scores = np.zeros((len(output_dims), 2))

  for i, output_dim in enumerate(output_dims):
    X_lsa = reduce_lsa(X, output_dim)
    X_doc2vec = reduce_doc2vec(X, output_dim)

    quality_scores[i][0] = fit_logistic(X_lsa, y)
    quality_scores[i][1] = fit_logistic(X_doc2vec, y)

  return quality_scores
```

Here's how we call it, along with D mentioned earlier.

```
# Get quality scores
output_dims = [i for i in range(50, 501, 10)]
quality_scores = get_quality_scores(X_train, y_train, output_dims)
```

With that done, we can now perform our hypothesis test. `scipy.stats` graciously implements this for us so we don't have to.

```
# Hypothesis test
statistic, p_value = wilcoxon(
    quality_scores[:, 0],
    quality_scores[:, 1],
```

```
)

print("Wilcoxon Signed-Rank Test:")
print("Test Statistic:", statistic)
print("p-value:", p_value)

alpha = 0.05
if p_value < alpha:
  print("Reject H_0")
else:
  print("Do not reject H_0")
```

5.4 Interpreting the results

Before we interpret the result of the hypothesis test, let's first look at the quality score samples. After about 2 hours of training, here's what I got:

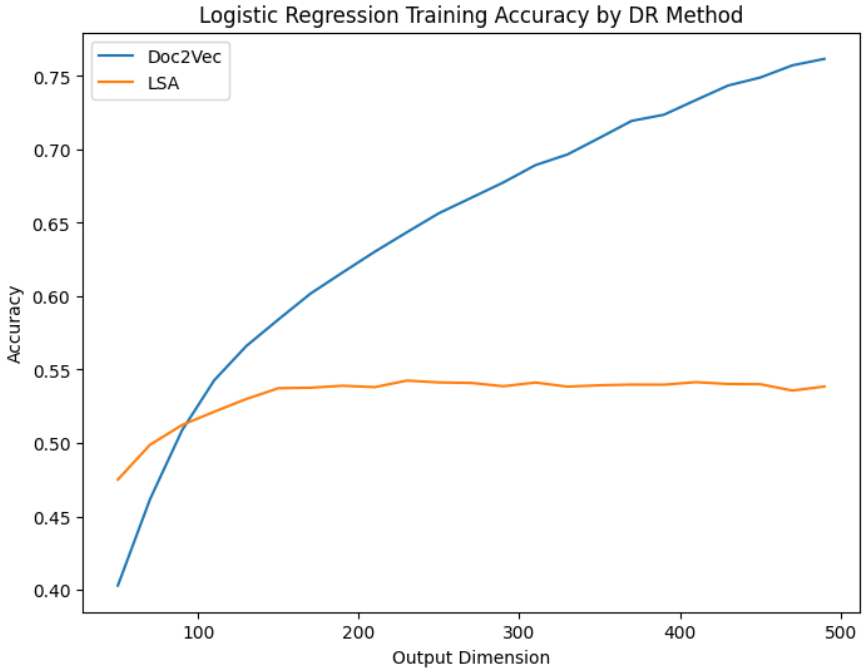

Figure 1: $Q_{\cdot 1}$ and $Q_{\cdot 2}$ w.r.t. D

Empirically it seems that Doc2Vec perform much worse for lower d_{reduced} and much better for higher d_{reduced} w.r.t. LSA. Indeed, this observation is echoed in the literature [9]. Now let's look at the hypothesis test results.

In this case, it's no surprise that H_0 was rejected (oops spoiler, it was). In other cases where g^1 and g^2 perform much more similarly, perhaps the test may have more utility.

This brings us to the concept of scientific significance v.s. statistical significance. Did we really have to perform a statistical hypothesis test to come to this conclusion? Isn't the conclusion obvious? It's worth mentioning that the result of our hypothesis test is very dependent on the D that we select. Perhaps we're only interested in $2 \leq d_{\text{reduced}} \leq 100$. In this case, it seems we would not reject H_0. The hypothesis test simply serves to answer a question we give it — it's still up to us to specify the question correctly.

6 Conclusion

In this project we demonstrate the application of Wilcoxon's signed-rank test for paired samples on quality score samples for pairwise comparison of dimensionality reduction algorithms. We outline the task of dimensionality reduction (DR), various DR algorithms, quality measures, and the task of DR algorithm comparison. We also review Wilcoxon's signed-rank test as taught in class, including its assumptions and use cases. Finally, we implement and analyze the methodology on the real-world task of identifying performant document embeddings for topic classification.

7 References

[1] Peter Bühlmann and Sara Van De Geer. *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media, 2011.

[2] Jiahua Chen. *STAT 460/560 + 461/561: Statistical Inference I & II*. 2023/2024.

[3] Asir Antony Gnana Singh Danasingh, Appavu alias Balamurugan Subramanian, and Jebamalar Leavline Epiphany. "Identifying redundant features using unsupervised learning for high-dimensional data". In: *SN Applied Sciences* 2.8 (2020), p. 1367.

[4] Ilias Diakonikolas et al. "Robust estimators in high-dimensions without the computational intractability". In: *SIAM Journal on Computing* 48.2 (2019), pp. 742–864.

[5] Antonio Gracia et al. "A methodology to compare dimensionality reduction algorithms in terms of loss of quality". In: *Information Sciences* 270 (2014), pp. 1–27.

[6] scikit-learn contributors. *5.6.2. The 20 newsgroups text dataset*. `https://scikit-learn.org/0.19/datasets/twenty_newsgroups.html`. 2024.

[7] Michael C Thrun, Julian Märte, and Quirin Stier. "Analyzing Quality Measurements for Dimensionality Reduction". In: *Machine Learning and Knowledge Extraction* 5.3 (2023), pp. 1076–1118.

[8] Michel Verleysen and Damien François. "The curse of dimensionality in data mining and time series prediction". In: *International work-conference on artificial neural networks*. Springer. 2005, pp. 758–770.

[9] Bin Wang et al. "Evaluating word embedding models: Methods and experimental results". In: *APSIPA transactions on signal and information processing* 8 (2019), e19.

[10] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. "Understanding bag-of-words model: a statistical framework". In: *International journal of machine learning and cybernetics* 1 (2010), pp. 43–52.