MACHINE LEARNING DEPARTMENT

Language Technologies Institute

Carnegie Mellon University

An Empirical Evaluation of Sketched SVD and its Application to Leverage Score Ordering

Hui Han Chin

Computer Science Department
Carnegie Mellon University

Paul Pu Liang

Machine Learning Department
Carnegie Mellon University

Presenter: Hui Han Chin

Two Motivating Questions

1. Is there a better way to generate mini-batch than random shuffle?
2. Do matrix sketching methods for leverage scores work in real life?
3. (We were taking a class by *the David Woodruff* of matrix sketching.)

Outline

1. Motivation
2. Empirical Evaluation of Sketched Leverage Scores
3. Sketched Leverage Score Ordering
4. Experiments of SLSO
5. Conclusion

Statistical Leverage Score

The **statistical leverage score** measures how much an outlier the data point is from other points in data matrix *A*, assuming a linear data model.

Naïve computation requires a costly matrix inversion. Fortunately, the techniques of **matrix sketching** can be used to approximate the leverage scores [Drineas, 2011].

Algorithm 1 Exact Leverage Score by SVD

Input : Given $N \times D$ matrix A
Output : Leverage score of ith row as l_i
Compute SVD, $A = U\Sigma V^T$
Compute l_i from the first D columns of the ith row $l_i = |U_i|_2^2$

Algorithm 2 Approximation Leverage Score by Sketching

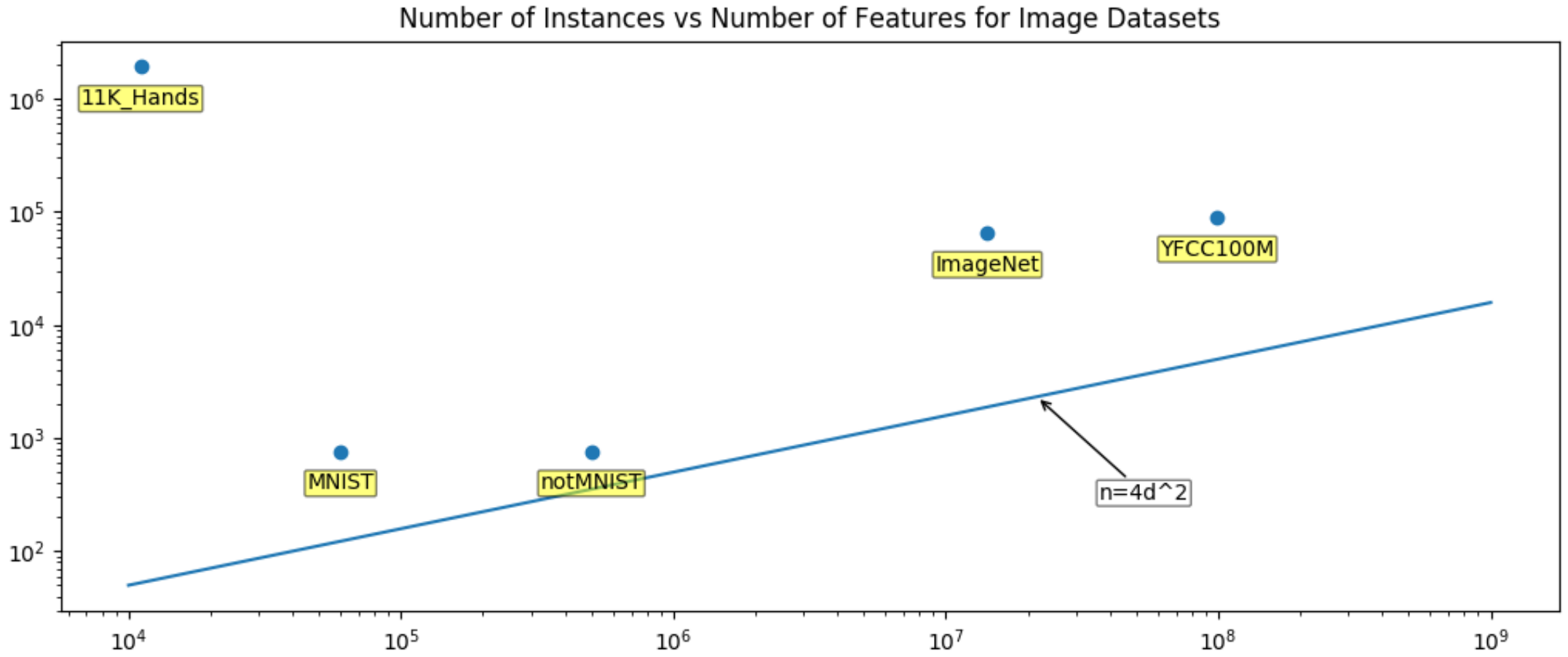
Input : Given $N \times D$ matrix A
Output : Approximate Leverage score of ith row as l_i
Compute Sketch of A, SA
Compute SVD, $SA = U\Sigma V^T$
Compute $U^{approx} = AV^T\Sigma^{-1}$
Compute l_i from the first D columns of the ith row $l_i = |U_i^{approx}|_2^2$

Sketching Techniques Evaluated

1. **Random Gaussian** [Woodruff, 2014]
2. **Fast Johnson Lindenstrauss transform** (FJLT) [Sarlos, 2006]
3. **Subsampled Randomized Hadamard Transform** (SRHT)
   [Boutsidis and Gittens, 2012]
4. **CountSketch** [Meng and Mahoney, 2013]
5. **OSNAP** [Nelson and Nguyen, 2013]

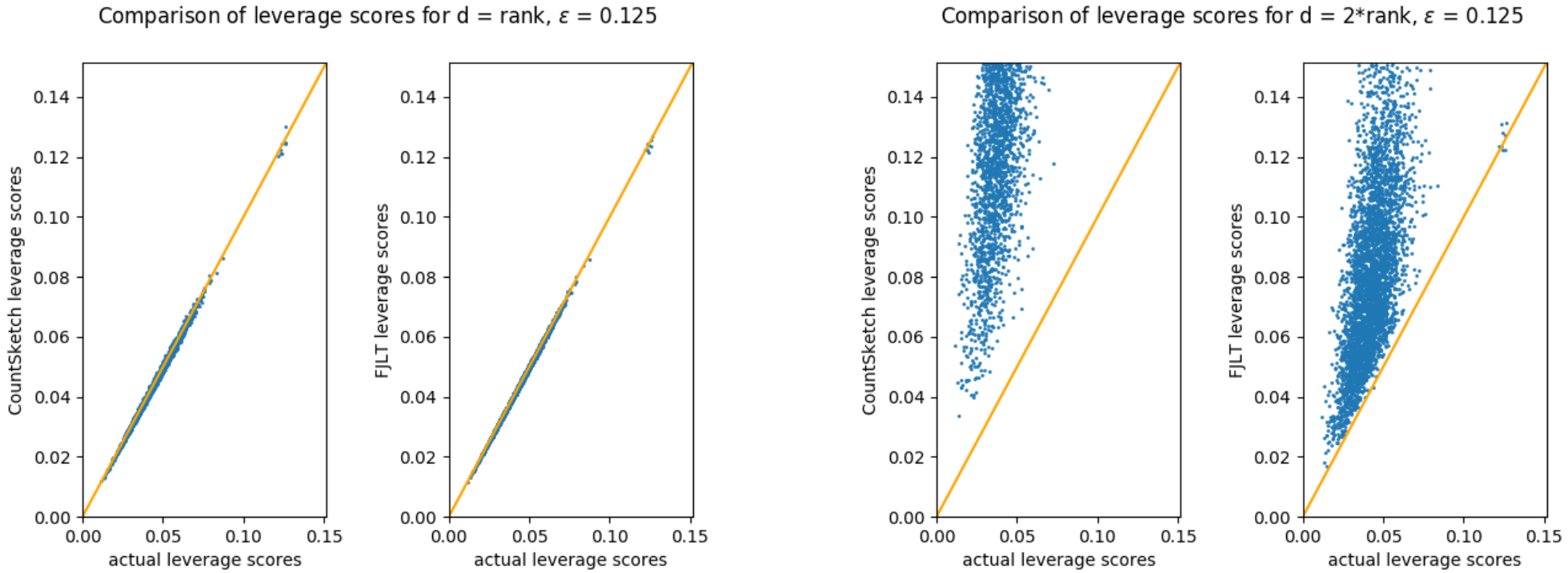
These sketching techniques have the ”subspace embedding property”

Empirical Evaluation of Sketched Leverage Score



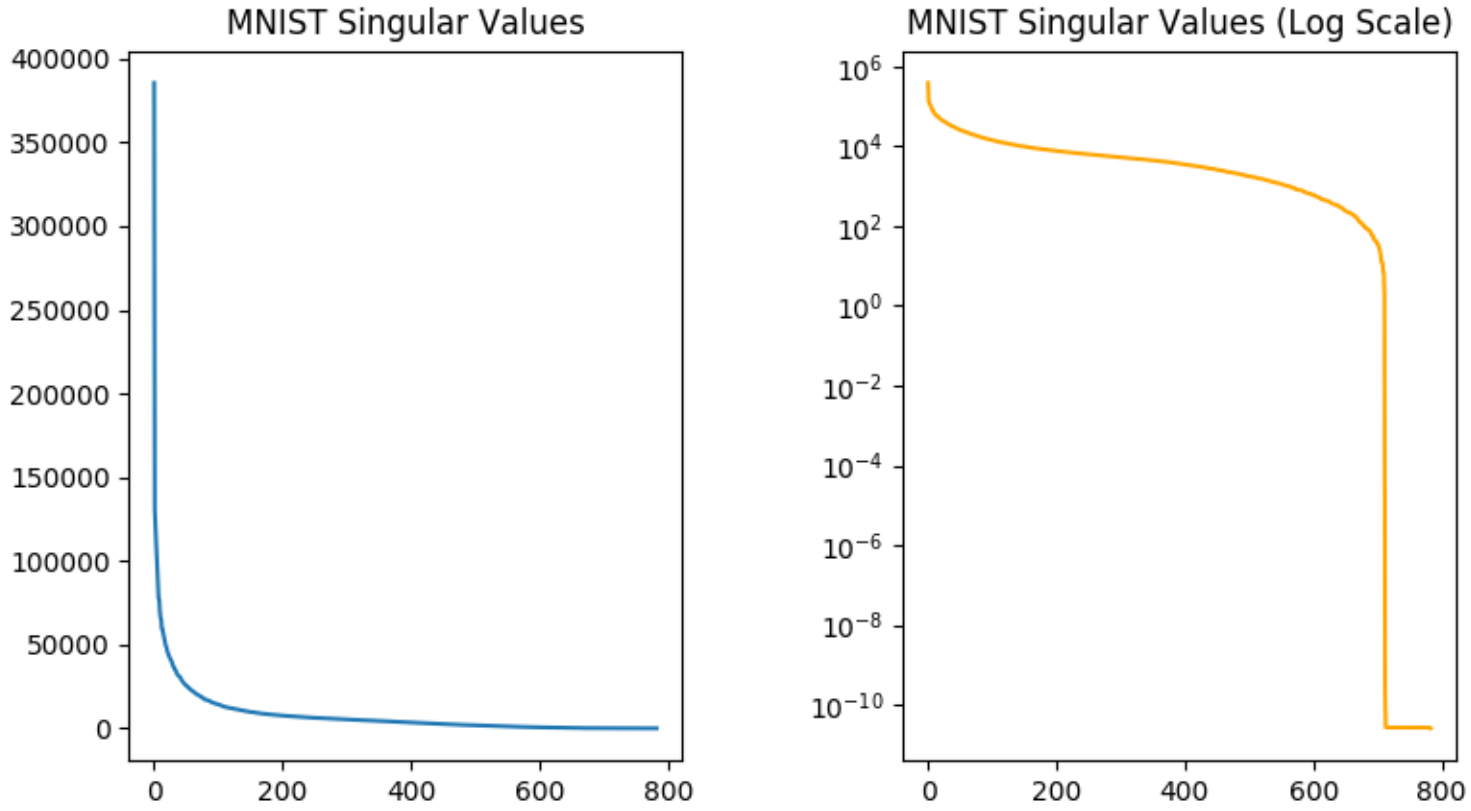
Most of the real world data sets do not meet the row-dim requirements for even CountSketch!

Empirical Evaluation of Sketched Leverage Score



- Quality of the sketch is highly dependent on the column rank of the data.
- Approximation error can be unbounded if the column rank < column dimension
- Requiring a full column rank matrix for sketching is a **hard requirement** [Clarkson and Woodruff, 2017].

Empirical Evaluation of Sketched Leverage Score



Most real data, like MNIST, have a high rank, low power noise.
=> The data "appears" full column rank

Insights from Sketched Leverage Scores

1. SRHT requirement of having rows as a power of 2 was impractical. The memory and timing overheads required to satisfy that requirement is prohibitive.
2. OSNAP is more useful for real world data as it requires less rows O(d).
3. Most curated real world datasets do not have enough rows (n) to satisfy the CountSketch (most analyzed) requirements.
4. Quality of the sketch is highly dependent on the column rank of the data. If the column rank of the data is less than the column dimension, then the approximation error can be unbounded.
5. Small singular values in real data can corrupt the approximate leverage scores returned by sketching. This happens often as real data have high rank noise.

Sketched Leverage Score Ordering

Approximate Leverage Score with Truncation

Input : Given $n \times d$ matrix $\mathbf{A}$, threshold ϵ.

Output : Approximate Leverage score of ith row as l_i.

1. Compute Sketch of $\mathbf{A}$, $\mathbf{SA}$.
2. Compute SVD, $\mathbf{SA} = \mathbf{U\Sigma V}^T$.
3. Truncate $\mathbf{V}, \mathbf{\Sigma}$ at small singular values less than ϵ. Let this truncated matrices be $\mathbf{V}'$, $\mathbf{\Sigma}'$
4. Compute $\mathbf{U}^{approx} = \mathbf{AV}'^T\mathbf{\Sigma}'^{-1}$.
5. Compute l_i from the first d columns of the ith row $l_i = |\mathbf{U}_i^{approx}|_2^2$.

Sketched Leverage Score Ordering

Using ideas from **curriculum learning**, the sketched leverage score is use to generate sampling policies to order the training data based on their leverage scores:

1. **Decreasing (dec)**:

Ordered based on strictly decreasing leverage scores. The most important and diverse training points are seen first.

2. **decreasing, sampling with replacement** (dec, swr):

The most important and diverse training points are seen first, but with randomness introduced.

3. **decreasing**, **sampling without replacement** (dec, swor):

Same but sampling without replacement.

4. **Shuffle** (Shuffle)**:**

baseline order where models are trained on shuffled training data.

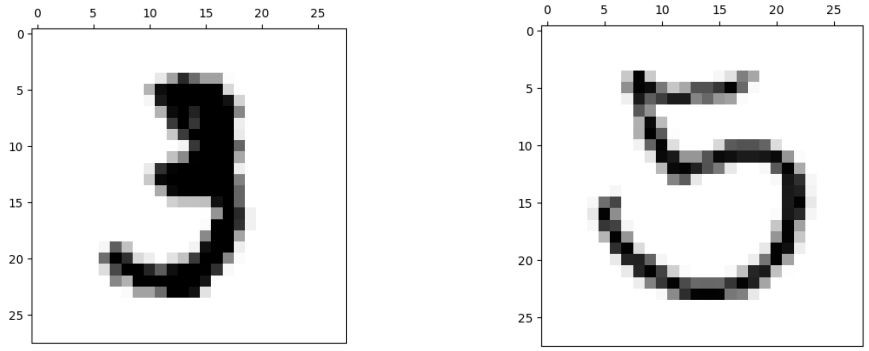
Experiments Setup

3 different datasets for Deep Learning. Models and hyper parameters kept constant across runs.

MNIST: image classification on digits handwriting.

SST: sentiment classification.

CMU-MOSI: multimodal sentiment analysis over 2 different runs using an early fusion method



Experiments Results

Task	**MNIST** 10 Class Image Classification Accuracy (%)				
Method	LR	NN Small	NN Large	CNN Small	CNN Large
shuffle	92.84	98.50	98.28	98.98	99.34
dec	89.09	98.43	98.34	**99.01**	98.99
dec, swr	92.70	98.42	98.46	**99.01**	99.35
dec, swor	**92.88**	**98.55**	**98.57**	**99.01**	**99.39**

MNIST, "decreasing order without sampling" improves performance.

Task	**SST** 5 Class Sentiment Classification Accuracy (%)				
Method	LR	DAN Small	DAN Large	LSTM Small	LSTM Large
shuffle	42.22	39.68	40.27	42.35	41.67
dec	**42.94**	39.86	**42.76**	42.35	41.31
dec, swr	41.22	**40.72**	40.72	**42.44**	**43.71**
dec, swor	42.26	39.41	40.14	41.27	41.36

SST, "decreasing order with replacement" improves performance.

Task	**CMU-MOSI** Sentiment Analysis											
Method	DAN Large				LSTM Small				LSTM Large			
Metric	A^2	F1	MAE	r	A^2	F1	MAE	r	A^2	F1	MAE	r
shuffle	61.2	59.9	1.314	0.438	73.9	74.0	1.068	0.624	73.3	73.3	1.067	0.604
dec	59.0	56.0	1.365	0.415	73.5	73.5	1.073	**0.626**	73.5	73.4	**1.038**	**0.621**
dec, swr	60.1	58.0	1.336	**0.413**	**74.6**	**74.7**	**1.061**	0.620	**74.1**	**73.9**	1.043	0.612
dec, swor	**64.3**	**64.3**	**1.271**	0.432	73.5	73.5	1.068	0.623	72.9	72.6	1.057	0.600

CMU-MOSI, "sampling in a decreasing order" improves accuracies.

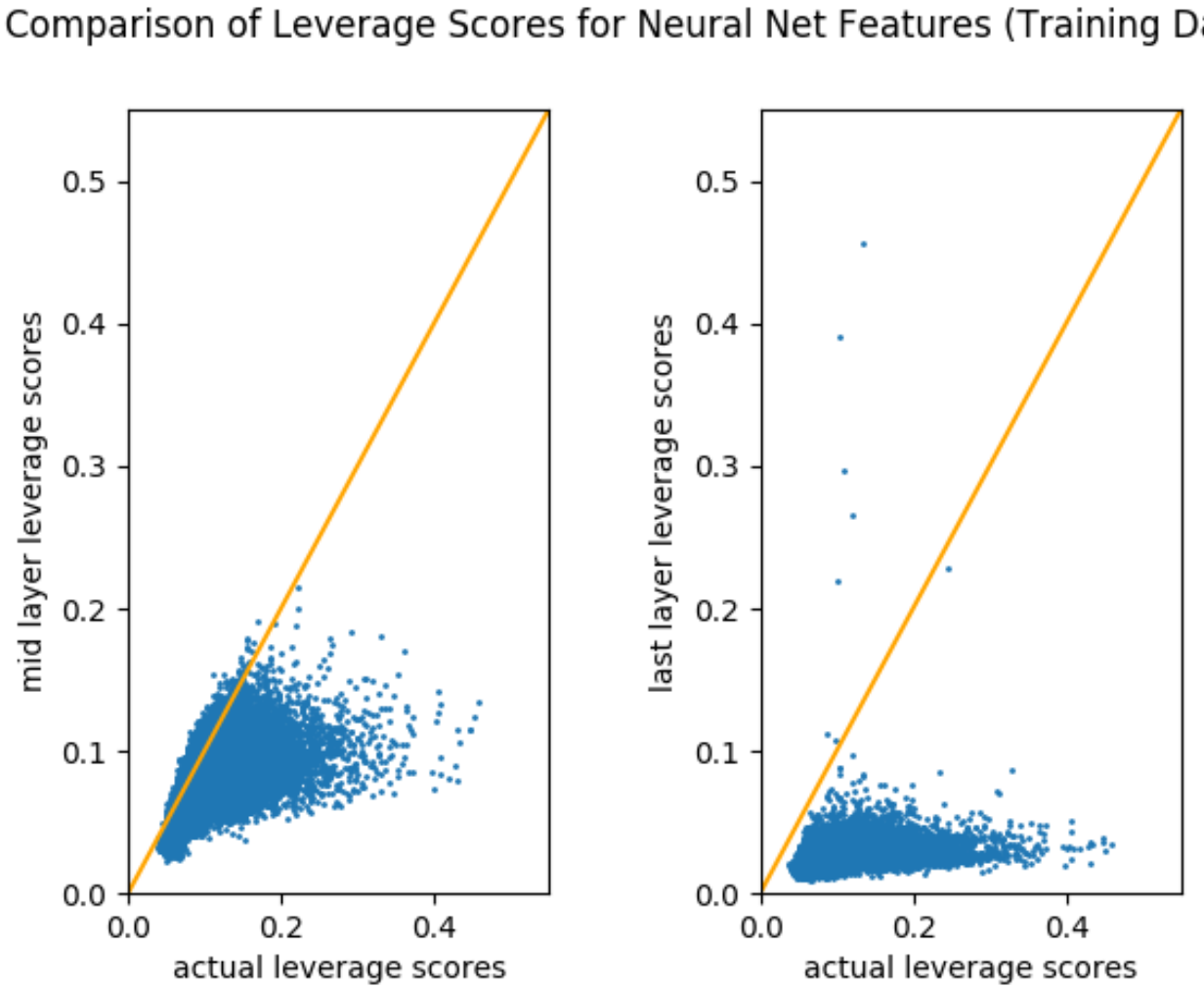
Decreasing SLSO improves performance compared to random shuffle!

Why Should this Even Work?

Curriculum Learning
manually determining a training order (Bengio et al., 2009; Spitkovsky et al., 2010; Khan et al., 2011)
Using auxiliary networks to jointly optimize for the optimal training order (Kumar et al., 2010; Jiang et al., 2015).

Solving Linear System
Randomized Coordinate Descent, Analysis of the Kaczmarz Method [Gower et al, 2015]

Why Should this Even Work?



MNIST data

Mid layer vs Last Layer

Two Motivating Questions

1. Is there a better way to generate mini-batch than random shuffle?

   Yes! SLSO is one approach.

2. Do matrix sketching methods for leverage scores work in real life?

   Only OSNAP is feasible for DL.

Conclusion

Empirical evaluation of Sketched Leverage Scores, identifying the limitations of matrix sketching to real world problems and guidelines for real world implementation.

Sketched Leverage Score Ordering, a technique for determining the ordering of data in the training of neural networks by computing leverage scores efficiently via truncated sketching.

Our method shows improvements in convergence and results.

Thanks for your attention!

Hui Han Chin
Email: hhchin87@gmail.com

Paul Pu Liang
Email: pliang@cs.cmu.edu
Twitter: @pliang279