Comparison of Stochastic Approximation Methods for PCA applied to a KNN Classifier

Corbin Rosset (crosset2, crosset2@jhu.edu), Edmund Duhaime (eduhaim1, eduhaim1@jhu.edu)

1 Abstract

In our research, weve discovered that training classifiers on high dimensional data sets with many examples is time the consuming. However, in the age of big data, this phenomenon is rapidly becoming the norm. When interpretability of a classifier is important to decision making (such as in the medical industry) it would be useful to have a low dimensional representation of the data that retains mostly the same information. There are known algorithms, such as Principal Component Analysis (PCA) and Independent Component Analysis (ICA) which transform high dimensional data in a lower dimensional feature space. The drawback to these classical algorithms is that their runtime is bounded by that of matrix multiplication and eigendecompositions, which may be prohibitive in some settings. Since we also want fast solutions, we would like linear time algorithms or $O(1)$ updates. Recently, many such algorithms have been proposed and studied. In our experiments, we intend to learn lower dimensional subspaces on our data using several incremental and stochastic approximations to PCA and its cousin Kernel PCA, and then train a simple KNN classifier on each subspace, and compare the performance. The algorithms we will use are normal eigendecomposition PCA, Kernel PCA, Power Iteration PCA, Stochastic Power Method (version of SGD), Incremental PCA, and Capped Matrix Stochastic Gradient (MSG).

2 Methods

The scope of our experiments is to test the performance of fast approximations to PCA. We do not wish to use fancy classifiers or ensemble algorithms, but surely those would lead to a more complete solution for production-ready code. Instead we will use a simple KNN. We will have two hyperparameters, the number of neighbors for KNN, and the number of principal components used. We will measure performance based on both time to train and the accuracy of the classifier. This is appropriate since we are interested in being able to both train quickly and to classify correctly. Stochastic approximation methods not only scale to large data sets, but they also apply to streaming data. Both KNN and stochastic PCA charge inexpensive incremental updates/predictions per example, making the pipeline suitable for real-time classification.

Also, the pairing of PCA and KNN is ideal because learning a lower dimensional subspace will mitigate the curse of dimensionality from which KNN can suffer under naive metrics such as euclidian distance. Also, the quality of the KNN prediction is a function of the quality of the learned subspace in which the training data lives: More informative principle components will hopefully lead to more distinct clusters.

3 Resources

For our data we will use the Extended Yale Face Dataset B
(http://vision.ucsd.edu/ leekc/ExtYaleDatabase/ExtYaleB.html) for testing purposes. This dataset includes thousands of pictures of 28 people in different poses and lighting conditions. Each picture has thousands of features, namely the pixels. This seems like a good dataset to use since we can classify based on person, something that KNN is useful for, and we can see how well that works with reducing the feature set from thousands to 20-30 features, so that KNN can do well. In addition we will develop on the Yale Face Dataset B that is a smaller set of different faces (http://vision.ucsd.edu/content/yale-face-database). We will implement the PCA algorithms from well-known publications. Because we have a need for fast linear algebra computations, we use Matlab. Matlab also has a built-in KNN library that we will use for all testing of KNN. Matlab will also be used to implement the various PCA algorithms, mainly due to its linear algebra library.

4 Milestones

4.1 Must achieve

A comparison of power iteration PCA and Stochastic Power Iteration on KNN classifiers with hyperparameter tuning to find the optimal neighbors and learned subspace dimension. Because it is known that power iteration converges arbitrarily close to the same solution as eigendecomposition-based PCA, we can use that to confirm our other algorithms are on the right track.

4.2 Expected to achieve

A comparison of the previously mentioned PCA methods on KNN in addition to incremental PCA, online PCA, and possibly capped MSG.

4.3 Would like to achieve

A comparison of previously mentioned PCA methods on other classifiers, such as SVM or decision trees. We would also like to try to add comparisons for other PCA methods if possible, such as online PCA. We would also like to try approximations to Kernel PCA as well. Also if possible we would like to try to add a normal eigendecomposition PCA, but this may not be possible due to limitations of what can be fit into memory.

5 Final Writeup

We will give a brief description of what each algorithm is and if applicable derive some performance bounds on the algorithm. Many of these algorithms are equipped with iteration bounds

for a given convergence threshold that depends on the eigengap between eigenvectors, for instance. We will directly compute and compare the cost of an incremental update for each algorithm. We will also discuss some caveats and tradeoffs of each algorithm. For example, Incremental PCA can fail under certain adversarially chosen inputs.

We can also discuss bias-variance tradeoff (dependent on the dimension of the learned subspace and the number of neighbors in KNN).

We will also plot classifier accuracy as a function of number of iterations for each of the algorithms (and hold the number of neighbors constant in KNN). We will also give the optimal values for the hyperparameters for each data set.

We will also briefly mention that often, only a handful of eigenvectors are needed to capture the vast majority of the variance. We may plot the eigen-decay graph (percent variance captured as a function of learned subspace dimensionality).

6 Bibliography

Arora, Raman, Andrew Cotter, Karen Livescu, and Nathan Srebro. "Stochastic optimization for PCA and PLS." In Allerton Conference, pp. 861-868. 2012.

Arora, Raman, Andy Cotter, and Nati Srebro. "Stochastic optimization of PCA with capped MSG." In Advances in Neural Information Processing Systems, pp. 1815-1823. 2013.

Coombe, Greg. "An introduction to principal component analysis and online singular value decomposition." graduate paper, University of North Carolina, Chapel Hill, NC (2006).

Warmuth, Manfred K., and Dima Kuzmin. "Randomized online PCA algorithms with regret bounds that are logarithmic in the dimension." Journal of Machine Learning Research 9, no. 10 (2008): 2287-2320.