Measuring the Effects of Dimensionality Reduction Algorithms: Principal Component Analysis on Optical Character Recognition Models with Support Vector Machine Classifier

Guy Nguyen-Phuoc

Cyber Security Research Laboratory
University of Hawaii - West Oahu

Abstract

Artificial Intelligence (AI) and Machine Learning (ML) have become synonymous with modern lifestyles. As such, large datasets are needed to train these models to receive accurate results with tradeoffs of increased computational costs. In this paper, we discuss the use of Principal Component Analysis to reduce the dimensions of these datasets while maintaining similar results to the original dataset. This is done through the UCI ML hand-written digits dataset with Support Vector Machines (SVM).

Keywords: Handwritten characters, support vector machines (SVM), optical character recognition (OCR), machine learning (ML), principal component analysis (PCA)

Introduction

Artificial Intelligence (AI) and Machine Learning (ML) are ubiquitous in modern society from the advertisements we see on the internet to how an Amazon.com order is shipped to our home [1]. AI/ML technology requires large amounts of data to make accurate predictions. Consequently, the amount of data in any AI/ML system generally increases its time to train the model. This makes training computationally expensive and requires specialized hardware to accelerate the training [2,3]. Principal component analysis (PCA) is a technique for reducing dimensionality while minimizing information loss [4]. We will use a popular dataset of handwritten characters with PCA to study its effects on a well-known Optical Character Recognition (OCR) model that uses the Support Vector Machine algorithm for feature extraction. In this paper, we will explore the use of dimensionality reduction algorithms such as PCA to address this issue of large dataset training times.

Background

A. Optical Character Recognition (OCR)

OCR is the use of electronic or mechanical processes to extract the printed or handwritten characters into a digital text format [5]. There are two types of OCR, offline and online. Among the offline versions, there are handwritten and machine-printed characters. This paper will focus on offline OCR, specifically handwritten characters. There are three steps in the OCR process: (i) Preprocessing, (ii) Feature extraction, and (iii) Classification. "The preprocessing methods include resizing, normalization, and the extraction of regions of interest (ROI)" [5]. The feature extraction step in the OCR process is used to extract features that will then be used as inputs for the classifier to determine the character image. There are many feature extraction methods, among them are: projection histograms, template matching, zoning, contour profiles, gradient descriptors, spline curve approximations, and Gabor features [6]. Feature extraction is extremely useful for OCR systems as it mitigates some issues of writing styles, size, and shape variations of handwritten characters. There are numerous types of classifiers available for OCR, from supervised methods such as decision trees, regression analysis, and naive Bayes to unsupervised methods that include k-means clustering [7]. However, for this paper the supervised SVM classifier method will be used.

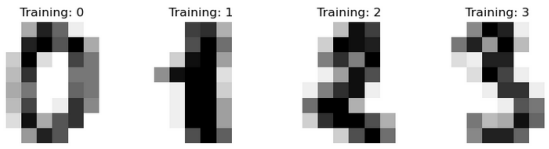

Fig. 1. Handwritten characters from UCI ML hand-written digits datasets.

B. Artificial Intelligence/Machine Learning

"Machine Learning (ML) is a branch of Artificial Intelligence (AI) that describes techniques through which systems learn and make intelligent decisions from available data" [8]. ML generally has three major groups of techniques: supervised learning, unsupervised learning, and reinforcement learning [7]. Additionally, there are two phases to ML called the training phase and inference phase. The training phase is described where the model undergoes learning through various ML techniques. Once the model completes the training phase it is tested on new data during the inference phase.

Table 1. Examples of Selected Machine Learning Techniques [8]

Machine Learning Techniques			
Supervised Learning		Unsupervised Learning	Reinforcement Learning
Classification	Regression	Clustering	Genetic Algorithms
SVM	SVR	HMM	Estimated Value
Naive Bayes	Linear Regression	GMM	Functions
K-NN	Decision Trees	k-means	Simulated Annealing
Discriminant Analysis	Ensemble Methods		
DNN	DNN		

Alzubi et al. describe a "Generic Model" for ML while recognizing that regardless of the ML technique there are six components that comprise the overall structure of an ML model [7].

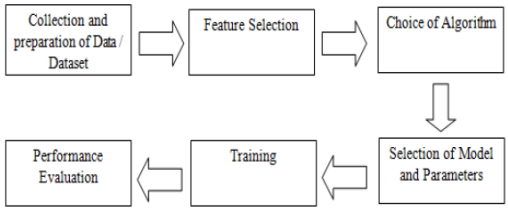

Fig. 2. Components of a Generic ML model [7]

The "Generic Model" process, as defined by Alzubi et al. is described as follows [7]:

I. Collection and Preparation of Data: Data is needed for any ML process, and it needs to be collected and prepared into an acceptable format that can be given to the ML algorithm as input. As such, large amounts of data may not be entirely relevant to the task, or it might contain noise that needs to be pruned. Data in this phase should be cleaned, normalized, and pre-processed into a usable format.

II. Feature Selection: Once the data is prepared, its features should be extracted. Depending on the data it may contain a variety of features, many of which are not necessary for the training step. This step should only take the most important features and prune the rest.

III. Choice of Algorithm: Some algorithms are better suited for certain tasks than other algorithms. This step is for choosing the best-suited algorithm for the task.

IV. Selection of Models and Parameters: Models rarely work out of the box and need to be tuned through various parameters to receive efficient results.

V. Training: Once the appropriate algorithms and parameters are set, the model needs to be trained. You should use only part of the available dataset for training to avoid over fitting the results.

VI. Performance Evaluation: After training, the model must be tested against the remaining unseen data to evaluate how much the model has learned. This is tracked with parameters like accuracy, precision, and recall.

C. Support Vector Machine

A Support Vector Machine (SVM) is a type of supervised ML algorithm based on Vapnik's statistical learning theory that is often used for object classification problems [8]. Supervised ML algorithms use sets of examples, defined by labels that are provided with the correct output for training. The accuracy of the model can be determined by

comparing the results of the model with the expected output of the dataset. This requires the labels in classification problems to have discrete values [7].

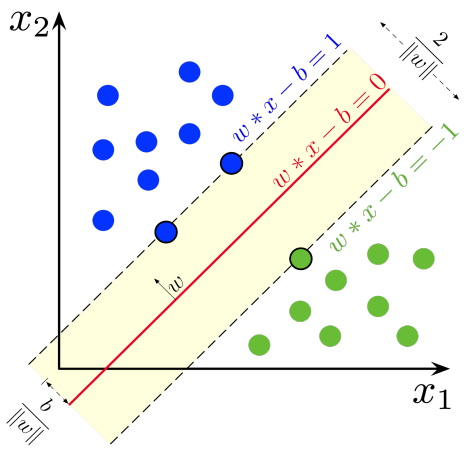

Fig. 3. An example of support vectors creating a linear relationship between two non-linear data points using an optimal hyperplane.

The general idea of an SVM is to take two different patterns and separate them based on the optimal hyperplane, seen in Fig. 3, in n-dimensional space [9]. The SVM model uses three lines to create the hyperplane:

w * x - b = 1
w * x - b = 0
w * x - b = -1

where w* x-b=0 is known as the margin of separation defined by the lines w * x - b = 1 and w * x - b = -1. The patterns that form on the edge of the hyperplane are called the support vectors where the perpendicular distance between the line of margin and the edges represents the margin [3]. The hyperplane is considered optimal when the set of hyperplanes maximizes the margin. Larger margin sizes indicate better pattern classifications. An example of an SVM equation is provided in Fig 4.

$$\max W(\propto) = \sum_{i=1}^{l} \propto_i - \frac{1}{2} \sum_{i=1}^{l} \cdot \sum_{j=1}^{l} y_i y_j k(x_i, x_j) \propto_i \propto_j$$

Fig. 4. Support vector machine equation [8].

D. Principal Component Analysis

Principal Component Analysis (PCA) is a dimensionality reduction algorithm that preserves the maximum amount of information, in the form of variance, while finding a lower-dimensional representation in the form of principal components [10]. This is accomplished by finding the sequences of orthogonal components that best fit the variance within the variables.

We can define the dataset X, where X is a part of Real numbers raised to the power of n columns (dimensions/features). This is projected into a lower-dimensional space X, where X is a part of Real numbers raised to the power of k columns, with k being fewer than n. PCA uses two phases, preprocessing and reduction to reduce the dimensions from n to k. The following 8 steps are defined by, Abdulhammed et al. [11].

1. Normalize the original data dimensions by using the equations below, where m is the number of instances in the dataset and X(i) are the data points.

$$\mu = \frac{1}{m} \sum_{i=1}^{m} X_{(i)}$$

2. Replace X(i) with X(i) - u.
3. Rescale the vectors Xj(i) to have unit variance using the equation below.

$$\sigma_j^2 = \frac{1}{m} \sum_{i} (X_{j(i)})^2$$

4. Replace Xj(i) with Xj(i)/σ.
5. Compute the Covariance Matrix Cov M with the equation below.

$$Cov_M = \frac{1}{m} \sum (X_{(i)})(X_{(i)})^\top$$

6. Calculate the Eigenvectors and corresponding Eigenvalues of Cov M.
7. Sort the Eigenvectors by decreasing the Eigenvalues and choose k Eigenvectors with the largest Eigenvalues to form W.

8. Use W to transform the samples onto the new subspace using the equation below

$$Cov_M = \frac{1}{m}\sum(X_{(i)})(X_{(i)})^\top$$

where X is a 1-dimensional vector and y is the transformed dimensional sample in the new subspace.

Experiments

This paper used the UCI ML hand-written digits datasets that were provided by the Scikit-learn dataset module [12]. Hyperparameters tuning was focused on the number of components for the PCA algorithm. The classifier was SVM, gamma, and probability parameters were used, set to "0.001" and "True" respectively. The model selection used random train/test splits set to "0.7" with shuffle set to "False". The "classifcation_report", "confusion_matrix", and "plot_confusion_matrix" that were provided by the Scikit-learn "metrics" module were used to graph the information through the "matplotlib" module.

Table 2. Algorithm Pseudo-code

Proposed Algorithm in Python

```
from sklearn import datasets, svm, decomposition, metrics
from sklearn.model_selection import train_test_split

X_digits, Y_digits = dataset.load_digits().data, dataset.load_digits().target

# Assume x starts at 1 and the upper bound is n - 1
For x in length(upperbound_n_components):
        X_train, X_test, Y_train, Y_test = train_test_split(
                decomposition.PCA(n_components=x).fit_transform(X_digits),
                Y_digits,
                test_size = 0.7,
                shuffle = False
        )
        print(f"{metrics.classification_report(
                Y_test,
                svm.SVC(gamma=0.001, probability=True).fit(X_train, Y_train).predict(X_test)
        )}")
```

Results

Table 3. Weighted Average Results

Weighted averages

N Components	Precision	Recall	f1-score	Accuracy
1.	0.35	0.37	0.35	0.37
2.	0.59	0.61	0.59	0.61
3.	0.71	0.72	0.71	0.72
4.	0.82	0.81	0.81	0.81
5.	0.87	0.87	0.87	0.87
6.	0.89	0.89	0.89	0.89
7.	0.91	0.91	0.91	0.91
8.	0.92	0.92	0.92	0.92
9.	0.92	0.92	0.92	0.92
10.	0.94	0.94	0.94	0.94
11.	0.94	0.94	0.94	0.94
12.	0.95	0.95	0.95	0.95
13.	0.95	0.95	0.95	0.95
14.	0.95	0.95	0.95	0.95
15.	0.96	0.96	0.96	0.96
*64.	0.96	0.96	0.96	0.96

* N = 64 means the model uses the complete data set
**Components 16 through 63 have been omitted due to them containing the same precision, recall, f1-score, and accuracy as component 15.

Confusion Matrix

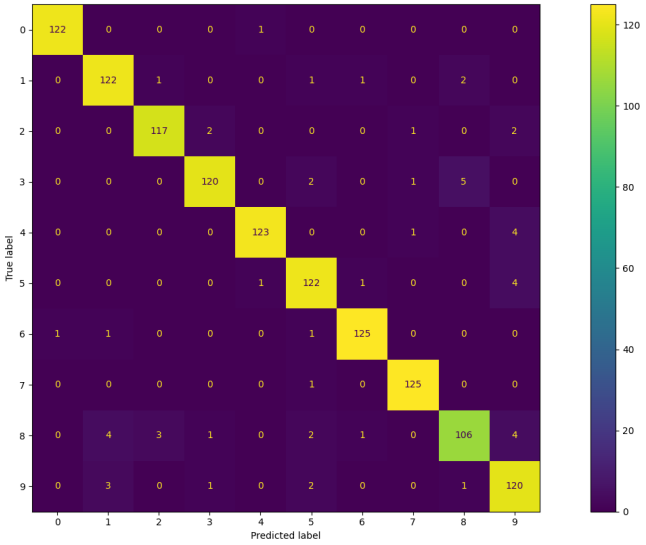

Fig. 5. Confusion Matrix for n=15 components.

The PCA of the handwritten dataset managed to reduce the overall size of the original dataset by 75% or 49 components. The resulting model, according to the table, matches the accuracy of the original dataset at just 15 principal components. The table above also demonstrates the diminishing returns of principal components, as by definition the first component

contains the most variance. Not all models need to match the original data set's accuracy when using dimensionality reduction methods such as PCA. The use of fewer components could be desirable if the accuracy of the model is sufficient for the model's use case.

Conclusion

This paper studied the use case of dimensionality reduction by way of PCA on handwritten characters. We were able to create a model that matches the accuracy of the original dataset with only 25% of the data or 15 principal components. While this preliminary result is promising, future work will be needed to assess if this approach is scalable on larger datasets such as the Modified National Institute of Standards and Technology (MNIST). MNIST is a large database containing nearly 70,000 images of handwritten digits commonly used for image processing systems. Hyperparameters were largely ignored throughout this paper to focus only on actively examining PCA. Parameters for the SVM and model selection, such as train split parameters, will be considered in future work. In addition, we would like to explore dimensionality reduction outside of OCR models.

References

[1] N. Maffulli et al., "Artificial intelligence and machine learning in orthopedic surgery: a systematic review protocol," Journal of Orthopaedic Surgery & Research, vol. 15, no. 1, pp. 1–5, Oct. 2020, doi: 10.1186/s13018-020-02002-z.

[2] J. Memon, M. Sami, R. A. Khan, and M. Uddin, "Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR)," IEEE Access, vol. 8, pp. 142642–142668, 2020, doi: 10.1109/ACCESS.2020.3012542.

[3] A. Pradhan, "Support Vector Machine-A Survey," IJETAE, vol. 2, no. 8, Aug. 2012.

[4] I. T. Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments," Phil. Trans. R. Soc. A., vol. 374, no. 2065, p. 20150202, Apr. 2016, doi: 10.1098/rsta.2015.0202.

[5] A. Sampath and N. Gomathi, "Fuzzy-based multi-kernel spherical support vector machine for effective handwritten character recognition," Sadhana, vol. 42, no. 9, pp. 1513–1525, Sep. 2017, doi: 10.1007/s12046-017-0706-9.

[6] P. Jayabala, E. Srinivasan, and S. Himavathi, "Performance analysis of hybrid feature extraction technique for recognizing English handwritten characters," Oct. 2012, pp. 373–377. doi: 10.1109/WICT.2012.6409105.

[7] J. Alzubi, A. Nayyar, and A. Kumar, "Machine Learning from Theory to Algorithms: An Overview," J. Phys.: Conf. Ser., vol. 1142, p. 012012, Nov. 2018, doi: 10.1088/1742-6596/1142/1/012012.

[8] T. S. Ajani, A. L. Imoize, and A. A. Atayero, "An Overview of Machine Learning within Embedded and Mobile Devices–Optimizations and Applications," Sensors (14248220), vol. 21, no. 13, pp. 4412–4412, Jul. 2021, doi: 10.3390/s21134412.

[9] D. A. Otchere, T. O. Arbi Ganat, R. Gholami, and S. Ridha, "Application of supervised machine learning paradigms in the prediction of petroleum reservoir properties: Comparative analysis of ANN and SVM models," Journal of Petroleum Science and Engineering, vol. 200, p. 108182, May 2021, doi: 10.1016/j.petrol.2020.108182.

[10] H. L. Shang, "A survey of functional principal component analysis," AStA Adv Stat Anal, vol. 98, no. 2, pp. 121–142, Apr. 2014, doi: 10.1007/s10182-013-0213-1.

[11] R. Abdulhammed et al., "Features Dimensionality Reduction Approaches for Machine Learning Based Network Intrusion Detection," Electronics, vol. 8, no. 3, p. 322, 2019, doi: http://dx.doi.org/10.3390/electronics8030322.

[12] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

This research was supported in part by the Department of Defense (DoD)/National Security Agency (NSA) Cybersecurity Scholarship Program (CySP) through a federal award under Grant Number H98230-20-1-0401.