Face Recognition in Machine Learning: A Framework for Dimensionality Reduction Algorithms

Professor Gabriel Kabanda
Adjunct Professor of Machine Learning
Woxsen School of Business, Woxsen University, Hyderabad, India
Email: Gabriel.Kabanda@woxsen.edu.in / gabrielkabanda@gmail.com
http://orcid.org/0000-0001-6699-080X

ABSTRACT

A facial recognition system matches a human face from a digital image or a video frame against an authentic repository of faces or Eigenfaces subject to algorithmic performance and detection accuracy. Dimensionality reduction is a type of unsupervised learning for which input is images of higher-dimensional data and these images are represented with a lower-dimensional space. The purpose of the research paper is to evaluate the performance of Dimensionality Reduction algorithms for face recognition using different approaches of Machine Learning (ML). The research uses the Interpretivist Paradigm characterised by a subjectivist epistemology, relativist ontology, naturalist methodology, and a balanced axiology. The quantitative methodology with an experimental research design was used. The results of the experiment show that only selecting the top M eigenfaces reduces the dimensionality of the data, and that too few eigenfaces results in too much information loss, and hence less discrimination between faces. With increasing dimensionality, the amount of training instances needed rises exponentially (i.e., kd). The performance of the Dimensionality Reduction Algorithm is benchmarked against the Clustering, Bayesian, Genetic, Reinforcement Q-Learning and Reinforcement A3C Algorithms. The outcome of the research makes significant value-adding contributions to the future of advances in Big Data Analytics and ML.

Keywords - **Dimensionality Reduction Algorithms, Cybersecurity, Artificial Intelligence, Machine Learning, Deep Learning, Big Data Analytics, Facial Recognition.**

Date of Submission: Aug 06, 2022 Date of Acceptance: Sept 21, 2022

I. INTRODUCTION

Massive amounts of data are produced in the Internet of Things (IoT) age from a number of heterogeneous sources, such as mobile devices, sensors, and social media. The most disruptive technologies today include Big Data Analytics, Artificial Intelligence (AI) and Robotics, Machine Learning (ML), Cybersecurity, Blockchain Technology, and Cloud Computing. These technologies have a significant impact on societies and economies as well as how technical innovation develops and spreads in emerging markets, which are more likely to accept and adapt innovations produced elsewhere. The two basic features of machine learning are the automatic analysis of large data sets and the creation of models for the broad relationships between data (ML). Another definition of big data analytics is the use of machine-learning algorithms to analyze information patterns and their relationships in order to aid in the realization of important decisions [1]. Big data analytics is a field that focuses on devising methodical methods for mining and analyzing enormous datasets and amounts of data that are too large for traditional data-processing techniques to manage. Conventional relational database systems were unable to manage unstructured data. Data mining is the process of extracting significant (non-trivial, implicit, previously unidentified, and potentially fascinating) information or patterns from data in enormous databases. Organizations may effectively monitor and forecast their key performance indicators (KPIs) with the use of big data analytics, which leads to more informed strategy decisions.

1.1 Background

A method of recognizing or verifying a person's identification using their face is facial recognition. A facial recognition system is a technology capable of matching a human face from a digital image or a video frame against an authentic repository of faces or Eigenfaces. Algorithmic performance and detection accuracy affects the recognition stage in Machine Learning (ML) and Big Data Analytics (BDA) systems. A form of unsupervised learning called "dimensionality reduction" uses images of higher-dimensional data as input, and then represents those images in a lower-dimensional space, like the Principal Component Analysis (PCA) space. The purpose of the research paper is to evaluate the performance of Dimensionality Reduction Algorithms for face recognition using different approaches of ML. The key objective is to choose an optimum set of features of lower dimensionality to improve classification accuracy using both feature extraction and feature selection methods.

A facial recognition system, which is often used to verify users through ID verification services, works by identifying and measuring facial features from a given image. It may match a human face from a digital image or a video frame against a database of faces. Facial recognition is a way of identifying or confirming an individual's identity using their face, and is a category of biometric security. Facial recognition software can identify people in real time as well as in still photos and movies.

The two-dimensional (2D) face recognition approaches include the following:

1. Eigenface
   a) Information Theory approach
   b) Identify discriminating components
2. Feature analysis
   a) Localisation of features
   b) Distance between features
   c) Feature characteristics
3. Neural networks
   a) Back propagation techniques
   b) Better for detection and localisation than identification
4. Graph matching
   a) Construct a graph around the face
   b) Possible need for feature localisation
   c) Can include other data (colour, texture)
5. Fisherface
   a) Uses 'within-class' information to maximise class separation

However, 3D facial recognition increases accuracy and removes pose and lighting problems.

Computers are taught to learn via a method called machine learning (ML). Large-scale data analysis using machine learning (ML) is known for developing models of the broad relationships between the data. The three classes of ML, according to [2], are as follows:

i. *Supervised learning:* training examples are provided to the algorithms in the form of inputs labeled with the desired results;
ii. *Unsupervised learning:* the algorithms are given unlabeled inputs;
iii. *Reinforcement learning:* the inputs are action sequences, observations, and incentives.

The common classical ML algorithms include the following:

i. Logistic Regression (LR)
ii. Naive Bayes (NB)
iii. Decision Tree (DT)
iv. Nearest Neighbor (KNN)
v. Ada Boost (AB)
vi. Random Forest (RF)
vii. Support Vector Machine (SVM)

Data is compressed by dimensionality reduction into a less dimensional subspace while maintaining the majority of the crucial details. Dimensionality reduction is crucial, and it refers to a sort of unsupervised learning in which the input is a representation of higher-dimensional images in a lower-dimensional environment. To reduce the size of huge images for Principal Component Analysis (PCA) representation in the PCA space, dimension reduction is necessary. The reduced image's dimensional space is called PCA space. The photos are huge in the spatial domain (standard size), measuring 200 × 180 pixels. When projected to PCA in the PCA workspace, this is condensed to a single dimension of 1 x 50 pixels. Dimensionality reduction uses a facial recognition system which is often used to verify users through ID verification services, and this works by identifying and measuring facial features from a given image from a query face and matches it with the most appropriate image in the eigenface, a database of images, as shown on Fig. 1 below.

Query Face	Recognized face

Fig. 1: Mapping the query face and recognized face in the eigenface

In addition to the role of recognizing faces, facial detection enables the gadgets to recognize the presence of faces. How does a computer interpret our expression and mood, as shown on Fig. 2 below? The key issues with facial recognition systems are that they are less accurate than other biometrics, need a lot of storage, and demand high-quality photos.

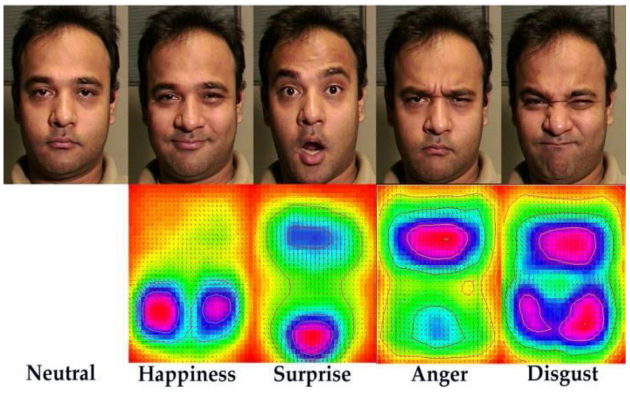


Fig. 2: Facial expressions that increase the complexity of facial recognition

1.2 Scope of the research

The goal of the research study is to compare the effectiveness of various ML methods for face recognition when used with Dimensionality Reduction Algorithms. The specific objectives are as follows:

a) To assess the performance of Classical Machine Learning Algorithms.
b) To develop a Cybersecurity system that uses ML paradigms.
c) To evaluate the performance of Dimensionality Reduction Algorithms for face recognition using different approaches of ML.
d) To determine an optimum set of features of lower dimensionality to improve classification accuracy using both feature extraction and feature selection methods.

II. LITERATURE REVIEW

The theoretical justification for dimensionality reduction comes from the viewpoint of statistics and probability. The Euclidean space has many wonderful characteristics in low-dimensional space, including the usage of the Euclidean distance metric for similarity computation and even the human visual system, which calculates distance in a Euclidean manner. The "curse of dimensionality" problem is a result of how these principles typically fail in high-dimensional areas and create odd and irritating events. Techniques for dimensionality reduction can be divided into two categories: feature transform and feature selection, depending on the justifications and methods of execution [3]. Dimensionality reduction has been the focus of extensive research both recently and for many years (some good overviews are available [4], [5],[6]. Today's information processing systems draw influence, in particular, from Sammon's pioneering work [6]. Sammon coupled dimensionality reduction with problems like classification and interactive visual data analysis in the late 1970s. Massive large-dimensional data generated by data mining, data warehousing, and knowledge discovery applications have recently rekindled interest in this subject. Effective dimensionality reduction techniques are also needed for other applications, including the genome project, text mining, and web mining. Unsupervised Learning explores the structure of data to extract meaningful information without the guidance of a known outcome variable or reward function. Clustering is a "unsupervised classification" technique that groups data into useful subgroups (clusters). Through dimensionality reduction, the data is compressed into a reduced dimensional subspace while keeping the majority of the crucial details.

Protecting the availability, confidentiality, and integrity of computing resources, networks, software, and data from attack requires a combination of rules, strategies, technologies, and processes [7]. [7] claims that cyberspace is a man-made information environment that is generated when computers, related telecommunications equipment, and other components that permit the quick transfer of enormous amounts of data are connected. The nonphysical aspect of cyberspace is revealed by the use of IP addresses. IP addresses provide navigational information to users without necessarily referring to a physical location, unlike addresses in the physical domain. Cyberspace is made up of interconnected gadgets and data that are all man-made. The physical layer, the intellectual layer, and the social layer are the three divisions of cyberspace.

The use of Machine Learning (ML), Neural Networks, and Fuzzy Logic to detect assaults on private networks was highlighted in research by [8] on the various Artificial Intelligence (AI) techniques. It should be emphasized that the majority of intrusion detection systems rely on signatures, making the development of a perfect, sophisticated intrusion detection system technically impossible. IDSs can be divided into host IDSs and network IDSs (NIDSs and NIDSs, respectively) (HIDS). Network-based IDS performs network traffic analysis and can identify unauthorized, illegal, and unusual behavior on the network.

III. RESEARCH METHODOLOGY

The Pragmatism paradigm used in this research is intricately related to the Mixed Method Research (MMR) and is a combination of positivism and interpretivism. The research method of mixed methods is largely quantitative with the research design being a survey and an experiment, and the Interpretivist Paradigm characterised by a subjectivist epistemology, relativist ontology, naturalist methodology, and a balanced axiology supported by qualitative approaches where Focus Group discussions were held.

The Principal Component Analysis (PCA) was used in exploratory data analysis and for making predictive models. PCA was used for dimensionality reduction. Face Recognition examples were illustrated by use of eigenfaces and the Support Vector Machine (SVM) algorithm implemented using Python programming. A comparison was made of the Dimensionality Reduction plot with the gallery of the most significative eigenfaces, from which it was concluded that the face recognition problem would be much more effectively solved by training convolutional neural networks.

The research also used the KDDCup 1999 intrusion detection benchmark dataset in order to build an efficient network intrusion detection system from which a Bayesian Network Model was developed. The study utilized the purposive sampling method. A sample of 494,020 instances were selected from the primary data, with about 10 million records and 42 attributes, obtained from http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

The Focus Groups were derived from 10 Groups of Masters students at the University of Zimbabwe in the 2022 cohort who conducted surveys and interviewed the management of various corporates in Zimbabwe and other neighbouring countries in Southern Africa.

IV. Data Analysis and Key Findings

We present and briefly discuss the most commonly used classical machine learning algorithms, according to [9], and the performance of these classical ML algorithms is illustrated below. We then present the key findings on the results of the Dimensionality Reduction Algorithms and compare them with other algorithms.

4.1 Classical Machine Learning (CML)

Computers can learn just like people thanks to a discipline of artificial intelligence called machine learning (ML). Below is a discussion of the most popular traditional machine learning algorithms.

4.1.1 Logistic Regression (LR)

Logistic regression is similar to linear regression in concept and was developed by [10], but it prevents misclassification that could happen in linear regression. Findings from logistic regression are essentially either "0" or "1," unlike results from linear regression. The quantity of training data has a significant impact on the effectiveness of logistic regression.

4.1.2 Naive Bayes (NB)

The Naive Bayes (NB) classifier is founded on the Bayes theorem, which presupposes feature independence. The Naive Bayes classifier escapes the dimensionality plague thanks to the independence assumptions.

4.1.3 Decision Tree (DT)

A decision tree has a structure similar to a flowchart, with the root node at the top and each internal node designating an aspect of the information. Due to the fact that even a small change in the information would alter the tree's structure, the algorithm may be biased and ultimately unstable.

4.1.4 K-Nearest Neighbor (KNN)

K-Nearest Neighbor (KNN) is a non-parametric method that employs similarity measures in relation to classifiers that use distance functions rather than news cases. KNN is computationally expensive since it saves all of the training data in a bigger amount of memory.

4.1.5 Ada Boost (AB)

A method for improving the efficiency of straightforward learning algorithms used for classification is the Ada Boost (AB) algorithm. Ada Boost combines a number of weak classifiers to create a strong classifier. It is a quick classifier that can also function as a feature learner simultaneously. This could be helpful for tasks involving the analysis of unbalanced data.

4.1.6 Random Forest (RF)

As an ensemble technique, random forest (RF) generates a decision tree from a subset of observations and variables. A single decision tree cannot predict as well as the Random Forest. It builds a number of minimally correlated decision trees using the bagging principle.

4.1.7 Support Vector Machine (SVM)

The supervised machine learning technology known as the Support Vector Machine (SVM) can be used to handle classification and regression issues. SVM is a linear classifier with a hyperplane as the classifier. The training set is separated by the widest possible margin. Support vectors are the places close to the dividing hype plane that dictate where the hyper plane will be.

4.2. Performance of selected Classical ML Algorithms

4.2.1. Classification and Regression Trees (CART)

The performance results of our CART algorithm in forecasting bank failure on the training set are displayed in Table 1 below. On the training dataset, the algorithm's accuracy rate was 82.8%. Our ideal model's best tuning or complexity parameter was 0.068. Our classifier was effective, as indicated by the Kappa statistic of 75% and the Kappa SD of 0.07 in the classification of bank categories. The algorithm's accuracy level and kappa on the test dataset were 92.5 percent and 88.72 percent, respectively. Only 2 instances were incorrectly categorized as moderate and 1 as satisfactory by the algorithm.

Complexity Parameter	Accuracy	Kappa	AccuracySD	KappaSD
0.06849315	0.8275092	0.7519499	0.04976459	0.07072572
0.15753425	0.7783150	0.6683229	0.07720896	0.14039942
0.42465753	0.5222344	0.1148591	0.08183351	0.18732422

TABLE 1: CART model performance.

Fig.3 below displays the accuracy of the CART model based on the complexity parameters of several test runs. The model performance was optimized by the complexity parameter, or the optimum tune parameter, of 0.068.

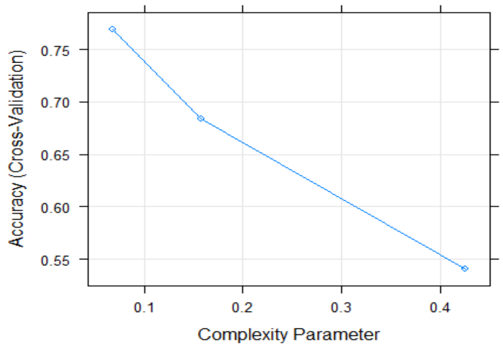


Fig.3: CART accuracy curve.

4.2.2 Support Vector Machine

According to Table 2, the SVM model's accuracy rate for forecasting bank solvency on the training dataset was 79.1%. Our extremely effective model's optimal tuning sigma and cost values were 0.05 and 1, as shown in Fig.4 below. The Kappa statistic and the Kappa SD were, respectively, 67.9 percent and 0.13. The algorithm's accuracy level and kappa on the test dataset were 92.5 percent and 88.54 percent, respectively. Comparing the method to the CART algorithm, the algorithm only misclassified 3 instances as moderate.

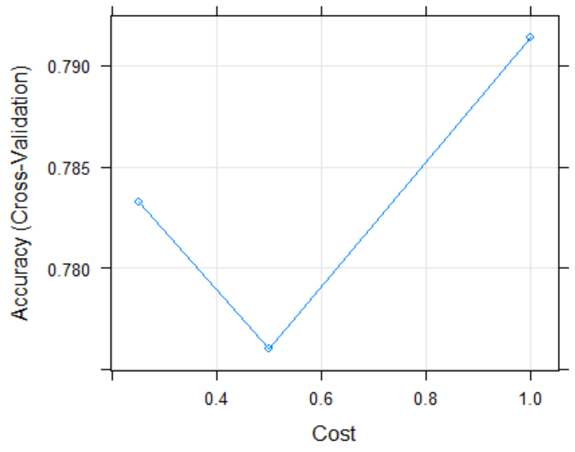


Fig.4: SVM accuracy curve

sigma	c	Accuracy	Kappa	AccuracySD	KappaSD
0.050398	0.25	0.783223	0.678536	0.095598	0.140312
0.050398	0.50	0.776007	0.661354	0.087866	0.132552
0.050398	1.00	0.791391	0.678694	0.080339	0.126466

TABLE 2: Support Vector Machine performance

4.2.3. Linear Discriminant Algorithm

Accuracy	Kappa	AccuracySD	KappaSD
0.8042399	0.7038131	0.1016816	0.159307

TABLE 3: Linear Discriminant algorithm performance

As shown in table 4, the LDA attained an accuracy level of 80% on the training dataset. Kappa SD was 0.16 and the Kappa statistic was 70%, respectively. The method had a kappa of 84.64 percent and a 90% accuracy level on the test dataset. Only 4 instances that performed poorly compared to the CART method were incorrectly labeled as moderate by the algorithm.

4.2.4. K-Nearest Neighbor

K	Accuracy	Kappa	AccuracySD	KappaSD
5	0.5988645	0.3698931	0.1280376	0.2158109
7	0.6268864	0.4072928	0.1564920	0.2703504
9	0.6621978	0.4715556	0.1747903	0.2881390

TABLE 4: K-NN algorithm performance

The training dataset's accuracy rate was 66.2 percent. According to the accuracy curve in Fig.5 below, k=9 or 9 neighbors was the ideal tuning parameter for our model. The Kappa statistic was 47.2 percent, and the Kappa SD was 0.17. The algorithm's accuracy level and kappa on the test dataset were 67.5 percent and 49 percent, respectively. Compared to other methods, the algorithm was not very good in classifying bank performance.

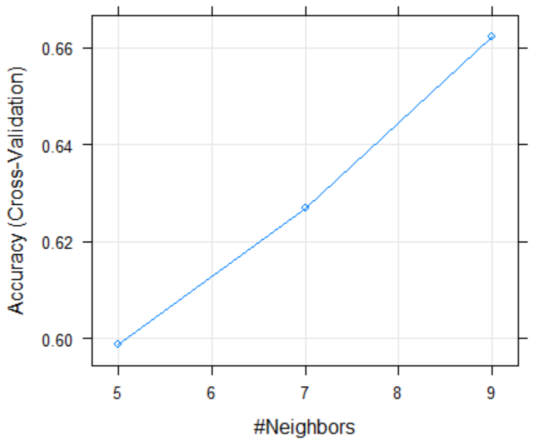


Fig. 5: K-NN confusion accuracy graph

4.2.5. Random Forest

mtry	Accuracy	Kappa	AccuracySD	KappaSD
2	0.8272527	0.7421420	0.10396454	0.15420079
14	0.8554212	0.7829891	0.06069716	0.09303130
16	0.8482784	0.7718935	0.06455248	0.09881991

TABLE 5: Random Forest performance

Our random forest's precision on the training set was 85.5 percent, as shown in Table 5. The number of predictors chosen at random for the purpose of building trees, as shown on Fig.6, was the optimal tuning parameter for our model, and it was 14. The Kappa statistic and the Kappa SD were, respectively, 78.3 percent and 0.09. The algorithm had a kappa of 96 percent and an accuracy level of 96 percent on the test dataset. Compared to other algorithms, the algorithm was quite good in classifying bank performance.

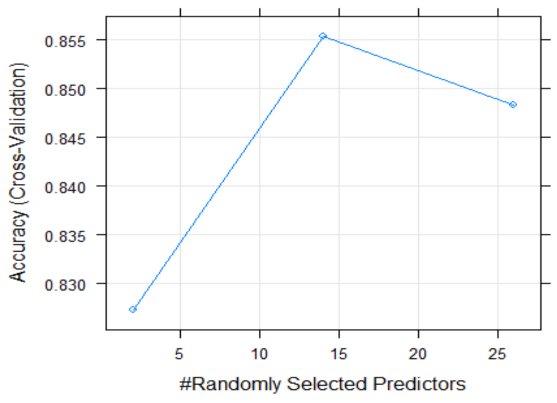


Fig. 6: Random forest accuracy graph

4.3. Dimensionality Reduction with Principal Component Analysis

By selecting only the top M eigenfaces, the dimensionality of the data is decreased. Due to insufficient eigenfaces, face discrimination is lacking, which results in severe information loss. Feature extraction involves taking a set of existing features and extracting new features from them using some sort of f() mapping. Selecting a subset of the initial features is known as feature selection. In actuality, adding additional features could result in lower performance. With increasing dimensionality, the amount of training instances needed rises exponentially (i.e., kd). Different techniques can be used to distinguish features between photos of faces using popular linear feature extraction methods: Principal Components Analysis (PCA) is an unsupervised learning approach that aims to project the data in a way that preserves as much of the data's original information as feasible. A projection that best discriminates the data is sought using linear discriminant analysis (LDA). There are many other methods that include Independent Component Analysis, Projection Pursuit, Isomap, Locally Linear Embedding, etc.

The Principal Component Analysis (PCA) was used in exploratory data analysis and for making predictive models. By projecting each data point onto just the first few principle components, PCA is frequently used to reduce the dimensionality of data while retaining as much of the variance as possible.

Suppose we are given x1, x2, ..., xM (N x 1) vectors for the case with N number of features an M data items. The key steps followed in the PCA were as follows:

❖ Step 1: compute sample mean

$$\overline{\mathbf{x}} = \frac{1}{M}\sum_{i=1}^{M}\mathbf{x}_i$$

❖ Step 2: subtract sample mean (i.e., center data at zero)

❖ $$\Phi_i = \mathbf{x}_i - \overline{\mathbf{x}}$$

❖ Step 3: compute the sample covariance matrix Σx

$$\Sigma_{\mathbf{x}} = \frac{1}{M}\sum_{i=1}^{M}(\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})^T = \frac{1}{M}\sum_{i=1}^{M}\Phi_i\Phi_i^T = \frac{1}{M}AA^T$$

where A=[Φ1 Φ2 ... ΦM], i.e., the columns of A are the Φi (N x M matrix)

❖ Step 4: compute the eigenvalues/eigenvectors of Σx

$$\Sigma_x u_i = \lambda_i u_i$$ where we assume $$\lambda_1 > \lambda_2 > \ldots > \lambda_N$$

❖ Step 5: dimensionality reduction step – approximate x using only the first K eigenvectors (K<<N) (i.e., corresponding to the K largest eigenvalues where K is a parameter):

$$\mathbf{x} - \overline{\mathbf{x}} = \sum_{i=1}^{N} y_i u_i = y_1 u_1 + y_2 u_2 + \ldots + y_N u_N$$

approximate using first K eigenvectors only

$$\hat{\mathbf{x}} - \overline{\mathbf{x}} = \sum_{i=1}^{K} y_i u_i = y_1 u_1 + y_2 u_2 + \ldots + y_K u_K$$

i. Face Recognition examples were illustrated by use of eigenfaces and the Support Vector Machine (SVM) algorithm using Python programming. A Dimensionality Reduction plot of the result of the prediction on a portion of the test set was done. This was compared with the Dimensionality Reduction plot done of the gallery of the most significative eigenfaces. From the images displayed, it was concluded that the face recognition problem would be much more effectively solved by training convolutional neural networks.
ii. The PCA implementation for Faces Recognition using Eigenfaces and SVMs was done in Python using the following example Python program.

```
from time import time
import matplotlib.pyplot as
plt
from sklearn.model_selection
import train_test_split
from sklearn.model_selection
import
RandomizedSearchCV
from sklearn.datasets import
fetch_lfw_people
from sklearn.metrics import
classification_report
from sklearn.metrics import
ConfusionMatrixDisplay
from sklearn.preprocessing
import StandardScaler
from sklearn.decomposition
import PCA
from sklearn.svm import
SVC
from sklearn.utils.fixes
import loguniform
```

```
lf w_people =
fetch_lfw_people(min_faces_per_perso
n=70, resize=0.4)
# introspect the images arrays to find
the shapes (for plotting)
n_samples, h, w =
lfw_people.images.shape
# for machine learning we use the 2 data
directly (as relative pixel
# positions info is ignored by this
model)
X = lfw_people.data
n_features = X.shape[1]
# the label to predict is the id of the
person
y = lfw_people.target
target_names =
lfw_people.target_names
n_classes = target_names.shape[0]
print("Total dataset size:")
print("n_samples: %d" % n_samples)
print("n_features: %d" % n_features)
print("n_classes: %d" % n_classes)
```

The resultant Dimensionality Reduction plot of the result of the prediction on a portion of the test set is shown on Fig.7 below.

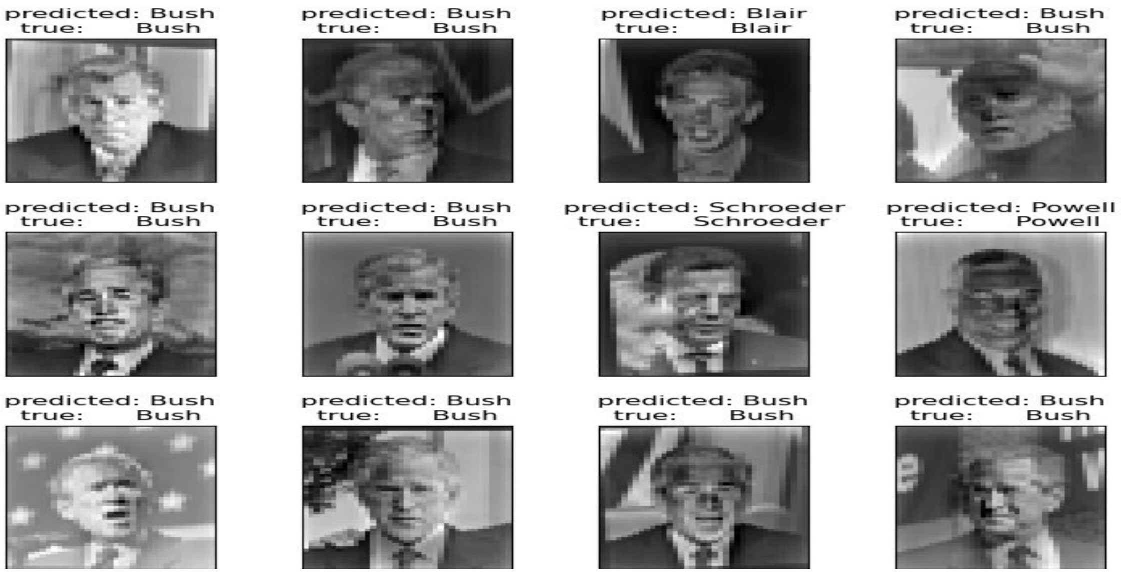


Fig.7: Dimensionality Reduction plot of the result of the prediction of the test set

Similarly, the Dimensionality Reduction plot of the gallery of the most significative eigenfaces used are shown on Fig.8 below.

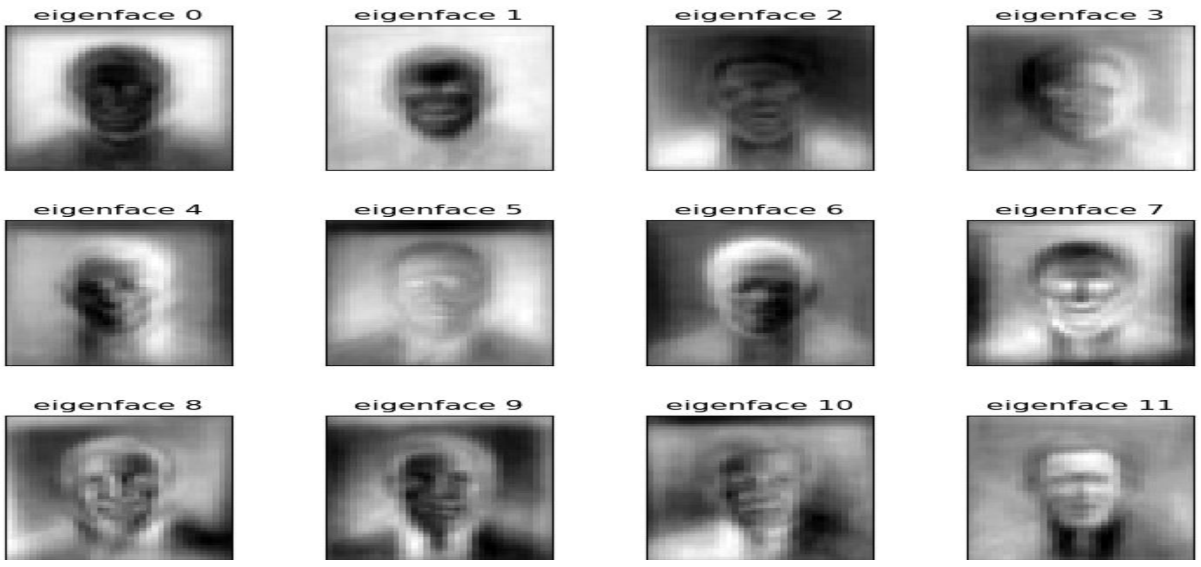


Fig.8: Dimensionality Reduction plot of the gallery of the most significative eigenfaces

From the results displayed on Fig.8, we conclude that the Face recognition problem would be much more effectively solved by training convolutional neural networks.

The results of the experiment show that only selecting the top M eigenfaces reduces the dimensionality of the data, and that too few eigenfaces results in too much information loss, and hence less discrimination between faces. Increasing the number of features will not always improve classification accuracy.

4.4. Bayesian Learning Learning

Cybersecurity is the practice of protecting systems, networks, and programs from digital (malicious) attacks. Cybersecurity is also the collection of policies, techniques, technologies, and processes that work together to protect the confidentiality, integrity, and availability of computing resources, networks, software programs, and data from attack [7]. Bayesian networks allow for prediction, generalization, and planning.

Bayesian Networks (BNs) are directed acyclic graphs that have an associated probability distribution function and these graphical probabilistic models are used for multivariate analysis [11].

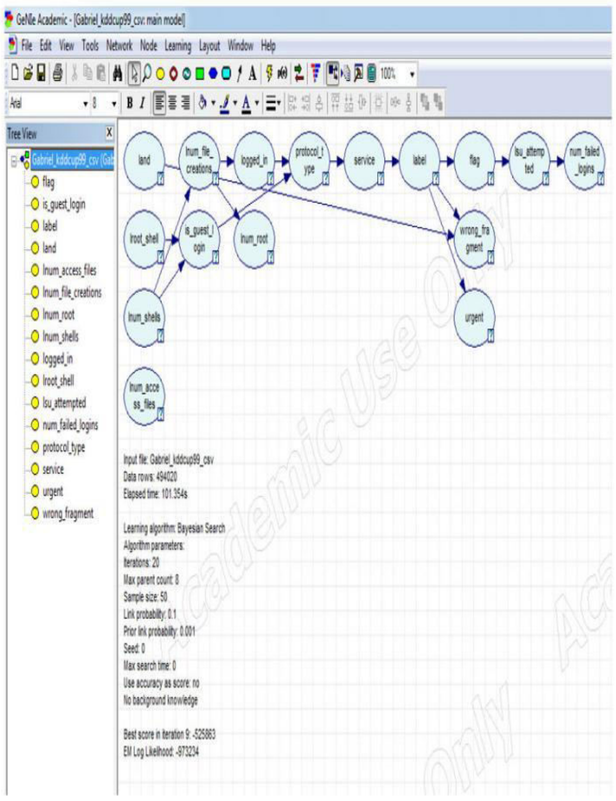
$$P(x) = {}^{n}\prod_{i=1} p(x_i | \Psi_i)$$

Allows us to reason from evidence to hypotheses.
Bayesian ML is a paradigm for constructing statistical models based on Bayes' Theorem and to compute the posterior probability

$$p(\theta, y) = p(\theta)\, p(y \mid \theta)$$

The research also used the KDD'Cup 1999 intrusion detection benchmark dataset in order to build an efficient network intrusion detection system from which a Bayesian Network Model was developed. A 95% confidence level (0.05) was used to calculate the sample size for the study and the study utilized the purposive sampling method. The primary data, with about 10 million records and 42 attributes, was obtained from
http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. However, a sample of 494, 020 instances were selected for data analysis. The Focus Groups were derived from 10 Groups of Masters students at the University of Zimbabwe in the 2021 cohort who then were tasked to conduct surveys and interview the management of various corporates in Zimbabwe and other nearby Southern African countries.

Following work done by [1], a Bayesian Network Model was developed. The SNORT open source software and other Bayesian Network supportive platforms such as NCSS 2019, Pass 2019, GeNIe 2.3, WinBUGS14, BayES and Analytica 5.1, were used to analyse the quantitative data. The Bayesian Network structure was developed as shown on Fig.9 below.



```
Input file: Gabriel_kddcup99_csv
Data rows: 494020
Elapsed time: 101.354s

Learning algorithm: Bayesian Search
Algorithm parameters:
Iterations: 20
Max parent count: 8
Sample size: 50
Link probability: 0.1
Prior link probability: 0.001
Seed: 0
Max search time: 0
Use accuracy as score: no
No background knowledge

Best score in iteration 9: -525863
EM Log Likelihood: -973234
```

Fig.9: A Bayesian Network model developed.

The consequent strength of influence is shown on Fig. 10.

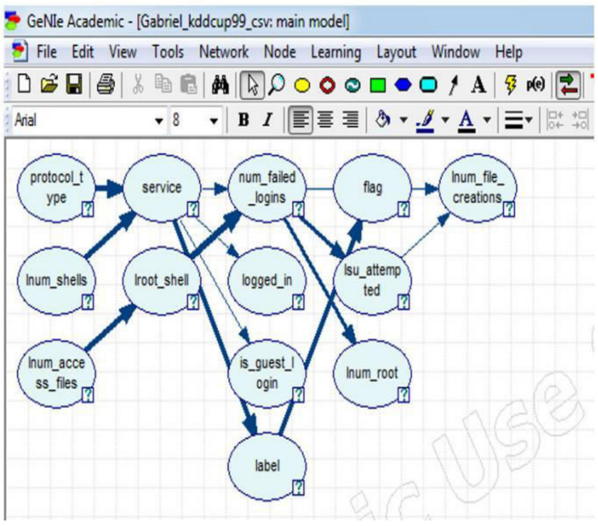


Fig. 10: Strength of Influence for the Bayesian Network Model

The benefits of using AI and machine learning in cyber security through the Bayesian Network model are as follows:
- ❖ Automated defense
- ❖ Faster defense and response
- ❖ Personalization
- ❖ Learn how to subtly adjust to the environment.
- ❖ Usability

4.5. Clustering Analysis and the K-means Algorithm

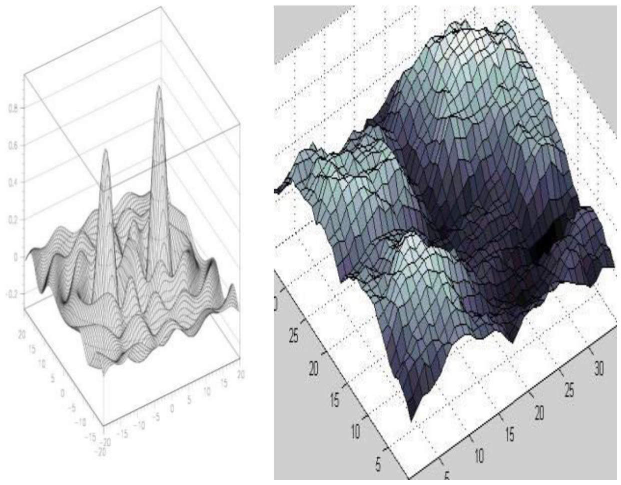
Clustering is the division of a data collection into subsets (clusters) so that the data in each subset (ideally) share some common attribute - frequently according to some defined distance measure. Clustering is the classification of items into different groups. The major clustering approaches include the following:
- ❖ *Partitioning algorithms:*
  Construct different partitions and then evaluate them according to a criterion
- ❖ *Hierarchical algorithms:* Create a hierarchical decomposition of the data set (or objects) according to a criterion
- ❖ *Density-based:* based on connectivity and density functions
- ❖ *Grid-based:* based on a multilevel granularity
- ❖ *Structure-Model-based:* A model is assumed for each of the clusters and the idea is to find the best fit of these models.

The k-means algorithm is a clustering algorithm that divides n objects into k subgroups based on qualities, where k < n. It is assumed that the object attributes form a vector space. This is a partitional clustering approach where each cluster is associated with a centroid (center point) and the number of clusters, K, must be specified. The key steps for the K-means Algorithm are as follows:

- Step 1: Choose k data points at random to serve as our starting centroids.
- Step 2: Using the k centroids, measure the distance (for our purposes, the Euclidean distance) between each data point in our training set.
- Step 3: Based on the calculated distance, place each data point next to its nearest centroid.
- Step 4: Update the centroid location by averaging each cluster group's points.
- Step 5: Continue Steps 2 through 4 until our centroids stay the same.

4.6. The Genetic Algorithm

Genetic Algorithms (GAs) are adept at navigating vast, possibly enormous search spaces in pursuit of ideal concoctions of variables—solutions you could not otherwise discover in a lifetime. GAs are directed search algorithms based on biological evolution's mechanics. An evolving population of potential answers to the problem is maintained using a genetic algorithm, which iteratively applies a collection of stochastic operators. GAs search the hypothesis space by generating successor hypotheses that repeatedly mutate and recombine elements of the best currently known hypotheses. GAs are a learning method that are motivated by analogy to biological evolution and provide efficient, effective methods for optimization and machine learning applications. The general format of the GA is as shown below.



```
{
initialize population;
evaluate population;
while
        TerminationCriteriaNotSatisfied
        {
          select parents for reproduction;
          perform recombination and mutation;
          evaluate population;
        }
}
```

Fig. 11: Genetic Algorithms fitness landscapes applicable to Face Recognition

The fitness landscapes for GAs that are applicable to facial recognition include the following examples shown on Fig.11.

4.7. Reinforcement Learning Algorithms

Reinforcement learning is learning from interactions with the environment in order to accomplish certain long-term objectives connected to the environmental condition. In reinforcement learning, the unlabeled raw data (or input features) is used. The reward signal, which must be maximized, defines the objective. Before the learning process ever begins, the reward system is coded. Through trial and error, the computer discovers patterns, and a human expert rewards it as necessary. The agent must be able to partially or fully sense the environment's state and act to change it. A feature-vector is often used to characterize the state. However, what is key is understanding how to act in a circumstance in order to maximize a numerical reward signal.

Asynchronous gradient descent is a technique for optimizing controllers in asynchronous reinforcement learning. This is helpful for deep reinforcement learning because deep neural networks, which are labor-intensive to train, are used as controllers.

The key Reinforcement Algorithms that have been tested and applied to face recognition systems are:

1) Asynchronous one-step Q-learning
2) Asynchronous one-step SARSA
3) Asynchronous n-step Q-learning
4) Asynchronous Advantage Actor-Critic (A3C)

The applicability of Reinforcement Algorithms, as illustrated by Asynchronous Algorithms, was demonstrated by the approach shown on Fig. 12 below.

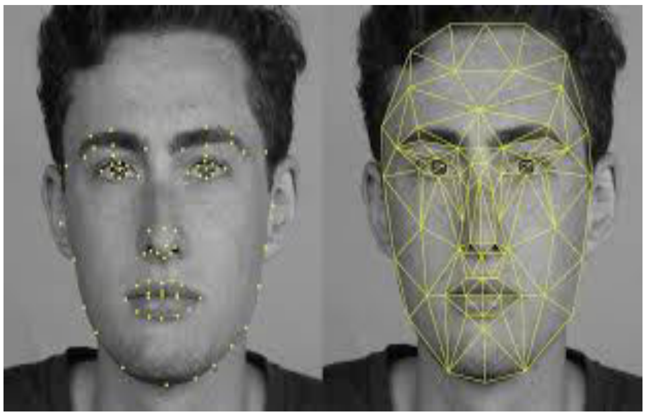


Fig. 12: Asynchronous Algorithms approach to Face Recognition

A reinforcement learning algorithm called *Q-learning* searches for the optimum course of action given the current situation. The goal of Q-learning is to discover a policy that maximizes overall reward. In q-learning, the "q" stands for quality. In this context, quality refers to how helpful a particular activity is in obtaining a potential reward. A model-free reinforcement learning algorithm called Q-learning is used to determine the worth of a given action in a given situation. The term "model-free" refers to the fact that it is not dependent on a model of the environment and that it is capable of handling stochastic transition and reward issues without the need for modifications.

The pseudo-code for the Asynchronous one-step Q-learning Algorithm is shown on Fig. 13.
Each thread engages with a separate copy of the environment and calculates the gradient of the Q-learning loss at each step. Utilize a target network with parameters that is widely shared and changes gradually. in calculating the Q-learning loss The target network used by DQN is also globally shared and gradually evolving; it is updated using a small batch of experience tuples taken from replay memory. The chance of many actor-learners overwriting each other's updates is decreased because the gradients are accumulated over a number of time-steps before being applied (akin to using mini-batches).

Algorithm 1: Asynchronous one-step Q-learning - pseudocode for each actor-learner thread.

```
// Assume global shared parameter vector θ.
// Assume global shared target parameter vector θ −.
// Assume global shared counter T = 0.
Initialize thread step counter t ← 0
Initialize target network weights θ − ← θ
Initialize network gradients dθ ← 0
Get initial state s
**repeat**
    Take action a according to the -greedy policy based on Q(s, a; θ)
    Receive new state s 0 and reward r y = r for terminal
    s 0 r + γ maxa 0 Q(s 0 , a0 ; θ −) for non-terminal s 0
    Accumulate gradients wrt θ: dθ ← dθ + $\partial(y-Q(s,a;\theta))^2$ ∂θ
    s = s 0
    T ← T + 1 and t ← t + 1
    **if** T mod Itarget == 0 **then**
        Update the target network θ − ← θ
    **end if**
    **if** t mod IAsyncUpdate == 0 or s is terminal **then**
        Perform asynchronous update of θ using dθ.
        Clear gradients dθ ← 0.
    **end if**
**until** $T > T_{max}$
```

Fig. 13: Asynchronous one-step Q-learning Algorithm

The n-step Q-learning Algorithm shown on Fig. 14 is a greater improvement to the one-step Q-learning algorithm. We have been utilizing one-step Q-learning thus far. One-step Q-learning changes the action value Q(s, a) towards the one-step return, r + maxa' Q(s', a'); Only the value of the state-action pair (s, a) that resulted in the reward is directly impacted by receiving a reward (r). Only indirectly are the updated values of the other state-action pairs impacted by the modified value Q (s, a). Due to the need for numerous updates to propagate a reward to the pertinent prior states and actions, this can slow down the learning process. To accelerate reward propagation, use n-step returns.

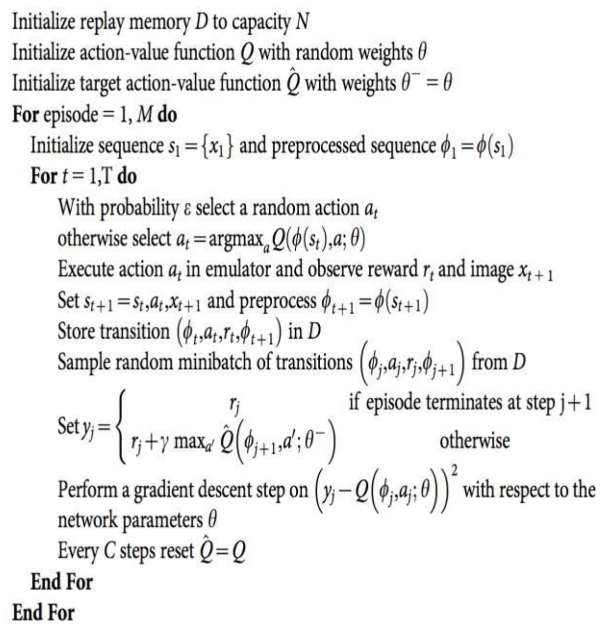
Algorithm 2: Asynchronous n-step Q-learning - pseudocode for each actor-learner thread.

```
// Assume global shared parameter vector θ.
// Assume global shared target parameter vector θ −.
// Assume global shared counter T = 0.
Initialize thread step counter t ← 1
Initialize target network parameters θ − ← θ
Initialize thread-specific parameters θ 0 = θ
Initialize network gradients dθ ← 0

**repeat**
    Clear gradients dθ ← 0
    Synchronize thread-specific parameters θ 0 = θ
    $t_{start}$ = t
    Get state $s_t$
    **repeat**
    Take action at according to the -greedy policy based on Q(st , a; θ 0 )
    Receive reward rt and new state st+1 t ← t + 1 T ← T + 1
    until terminal $s_t$ or $t - t_{start} == t_{max}$
    R = 0 for terminal $s_t$
    $\max_a$ Q(st , a; θ −)
    **for** non-terminal $s_t$ for i ∈ {t − 1, . . . , tstart} **do**
    R ← ri + γR
    Accumulate gradients wrt θ 0 : dθ ← dθ + $\partial(R-Q(si,ai;\theta 0))^2$ ∂θ0
    **end for**
    Perform asynchronous update of θ using dθ.
    **if** T mod Itarget == 0 **then**
        θ − ← θ
    **end if**
**until** $T > T_{max}$
```

Fig. 14: Asynchronous n-step Q-learning Algorithm.

Finally, Q-learning with experience replay shown on Fig. 15 perfects the Reinforcement Learning algorithms as applicable to face recognition.



```
Initialize replay memory $D$ to capacity $N$
Initialize action-value function $Q$ with random weights $\theta$
Initialize target action-value function $\hat{Q}$ with weights $\theta^- = \theta$
For episode = 1, $M$ do
  Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$
  For $t$ = 1,T do
    With probability $\varepsilon$ select a random action $a_t$
    otherwise select $a_t = \text{argmax}_a Q(\phi(s_t), a; \theta)$
    Execute action $a_t$ in emulator and observe reward $r_t$ and image $x_{t+1}$
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $D$
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $D$
    Set $y_j = \begin{cases} r_j & \text{if episode terminates at step } j+1 \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^-) & \text{otherwise} \end{cases}$
    Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ with respect to the
    network parameters $\theta$
    Every $C$ steps reset $\hat{Q} = Q$
  End For
End For
```

Fig. 15: Q-learning with experience replay

In conclusion, the Asynchronous Advantage Actor-Critic (A3C) Algorithm shown on Fig. 16 is one of the best Reinforcement Algorithms. A3C performs better on Deep Reinforcement Learning challenges and is quicker and easier to use. It maintains a policy π(at| st; θ) and an estimate of the value function V(st; θv). A3C uses parallel actor-learners and collects updates, just like with value-based approaches, to enhance training stability. In reality, several parameters are shared between the value function and the policy approximation, that includes the use of a neural network with shared non-output parameters that has one softmax output for the policy π(at| st; θ) and one linear output for the value function V(st; θv).

Algorithm 3: Asynchronous advantage actor-critic - pseudocode for each actor-learner thread.

```
// Assume global shared parameter vectors θ and θv and global shared counter T = 0 //
Assume thread-specific parameter vectors θ 0 and θ 0 v
Initialize thread step counter t ← 1
repeat
    Reset gradients: dθ ← 0 and dθv ← 0.
    Synchronize thread-specific parameters θ 0 = θ and θ 0 v = θv
    $t_{star}t$ = t
    Get state st
    repeat
        Perform $a_t$ according to policy π(at |st ; θ 0 )
        Receive reward $r_t$ and new state st+1
        t ← t + 1
        T ← T + 1
    until terminal $s_t$ or t − $t_{start}$ == $t_{max}$
    R = 0 for terminal $s_t$
    V (st , θ0 v ) for non-terminal $s_t$// Bootstrap from last state
    for i ∈ {t − 1, . . . , $t_{star}t$} do
        R ← $r_i$ + γR
        Accumulate gradients wrt θ 0 : dθ ← dθ + ∇ θ 0 log π(ai |si ; θ 0 )(R − V (si ; θ 0 v ))
        Accumulate gradients wrt θ 0 v : dθv ← dθv + ∂ (R − V (si ; θ 0 v ))2 /∂θ0 v
    end for
    Perform asynchronous update of θ using dθ and of θv using dθv.
until T > $T_{max}$
```

Fig. 16: Asynchronous Advantage Actor-Critic (A3C) Algorithm

V. Conclusion

Algorithmic performance and detection accuracy affects the recognition stage in Machine Learning (ML) and Big Data Analytics (BDA) systems. Dimensionality reduction is a type of unsupervised learning for which input is images of higher-dimensional data and these images are represented with a lower-dimensional space such as the Principal Component Analysis (PCA) space. The purpose of the research paper was to evaluate the performance of Dimensionality Reduction Algorithms for face recognition using different approaches of ML.

The Pragmatism paradigm, a combination of positivism and interpretivism, was used with the research design being a survey and an experiment, and qualitatively Focus Group discussions were held. The Principal Component Analysis (PCA) was used in exploratory data analysis and for making predictive models in the dimensionality reduction.

Face Recognition examples were illustrated by use of eigenfaces and the Support Vector Machine (SVM) algorithm implemented using Python programming. The results of the experiment show that only selecting the top M eigenfaces reduces the dimensionality of the data, and that too few eigenfaces results in too much information loss, and hence less discrimination between faces. The performance of the Dimensioanality Reduction Algorithm was compared against and complemented with the Clustering Algorithm, Bayesian Algorithm, Genetic Algorithm, Reinforcement Q-Learning Algorithm and Reinforcement - A3C Algorithm.

References

[1]. KABANDA, G., (2020), Performance of Machine Learning and other Artificial Intelligence paradigms in Cyber security, *Oriental Journal of Computer Science and Technology*, May, 2020,*Volume 13*, Issue Number 1 of 2020, pages 1-21, ISSN : 0974-6471 , Online ISSN : 2320-848 , http://www.computerscijournal.org/vol13no1/performance-of-machine-learning-and-other-artificial-intelligence-paradigms-in-cybersecurity/

[2]. TRUONG, T.C; Diep, Q.B.; & Zelinka, I. (2020), Artificial Intelligence in the Cyber Domain: Offense and Defense, *Symmetry* 2020, *12*, 410.

[3]. X. He, S. Yan, Y. Hu, P. Niyogi, and H. Zhang, (2005), Face recognition using Laplacianfaces,*IEEE Trans. Pattern Analysis and Machine Intelligence*, *vol. 27*, no. 3, pp. 328-340, 2005.

[4]. DUNTEMAN, G. H., (1989). *Principal Components Analysis*. Sage Publications, 1989.

[5]. FOLEY, D. and J. W. Sammon. (2015), An optimal set of discriminant vectors. *IEEE Transactions on Computers*, *24 (5)*:281–289, 2015.

[6]. SAMMON, J. W., (1969), A non-linear mapping for data structure analysis,*IEEE Transactions on Computers*, *C-18* (5):401–409, 1969.

[7]. BERMAN, D.S., Buczak, A.L., Chavis, J.S., and Corbett, C.L. (2019),Survey of Deep Learning Methods for Cyber Security, *Information* **2019**, *10*, 122; doi:10.3390/info10040122.

[8]. NAPANDA, K., Shah, H., and Kurup, L., (2015), Artificial Intelligence Techniques for NetworkIntrusion Detection, *International Journal of Engineering Research & Technology (IJERT),* ISSN: 2278-0181, IJERTV4IS110283 www.ijert.org,*Vol. 4,* Issue 11, November-2015.

[9]. KABANDA, G., (2021), Performance of Machine Learning and Big Data Analytics paradigms in Cybersecurity and Cloud Computing platforms, *Global Journal of Computer Science and Technology: G Interdisciplinary, Volume 21*, Issue 2, Version 1.0, Year 2021, pages 1-25; Type: Double Blind Peer Reviewed International Research Journal; Publisher: Global Journals Online ISSN: 0975-4172 & Print ISSN: 0975-4350; DOI : 10.17406/GJCST; Performance of Machine Learning and Big Data Analytics Paradigms in Cybersecurity and Cloud Computing Platforms (globaljournals.org); Performance of Machine Learning and Big Data Analytics paradigms in Cybersecurity and Cloud Computing Platforms | Global Journal of Computer Science and Technology (computerresearch.org).

[10]. SITI Nurul Mahfuzah, M., Sazilah, S., & Norasiken, B. (2017), An Analysis of Gamification Elements in Online Learning To Enhance Learning Engagement,*6th International Conference on Computing & Informatics*.

[11]. SARKER, I. H., Kayes, A. S. M., Badsha, S., Alqahtani, H., Watters, P., & Ng, A. (2020), Cyber security data science: an overview from machine learning perspective, *Journal of Big Data*. https://doi.org/10.1186/s40537-020-00318-5