An Efficient Hyperdimensional Computing Paradigm for Face Recognition

MOHAMMAD YASSER¹, KHALED F. HUSSAIN², AND SAMIA ABD EL-FATTAH ALI¹

¹Department of Electrical Engineering, Assiut University, Assuit 71515, Egypt
²Faculty of Computers and Information, Assiut University, Assuit 71515, Egypt
Corresponding author: Mohammad Yasser (mohammad_yasser@aun.edu.eg)

ABSTRACT In this paper, a combined framework is proposed that includes Hyperdimensional (HD) computing, neural networks, and k-means clustering to fulfill a computationally simple incremental learning framework in a facial recognition system. The main advantages of HD computing algorithms are the simple computations needed, the high resistance to noise, and the ability to store excessive amounts of information into a single HD vector. The problem of incremental learning revolves around the ability to regularly update the knowledge within the framework to include new subjects in an online manner. Using an HD computing classifier proved efficient and highly accurate to implement an incremental learning framework as no re-training was required after each online update to the framework which is HD computing biggest advantage. Another advantage is that HD computing classifiers can achieve a high degree of generalization. The framework was tested on a total of 11 open source benchmark data sets. A number of experimental tests were preformed to ensure consistent performance of the framework under different conditions against different data sets.

INDEX TERMS Computer vision, facial recognition, HD computing, incremental learning.

I. INTRODUCTION

Fast facial recognition algorithms have always been in high demand due to the need for reliable solutions on enterprise level e.g., within a corporation for personnel identification. The biggest challenge, when creating a facial recognition algorithm, is to implement the ability to update the dataset within the system regularly for it to recognize new subjects over time. The algorithm should also be implemented in a way that utilize the repetitive patterns and features from its training samples to reach higher efficiency. This challenge is referred to as incremental learning. The research field of incremental learning models has offered many algorithms over the years [1], [2].

There are multiple models in modern computing that take the brain’s circuits as a base for them. The most common example is neural networks with their flexibility and ability to learn a variety of datasets to solve many problems in computing in recent years pushing the progression of AI research in many different directions. One area that has had remarkable success with neural networks has been the field of facial recognition given the robustness of neural networks and their ability to extract features from images through different techniques. However, one frequent problem that facial recognition AI research with neural networks was the inability to update the neural networks with more subjects to classify easily. Training a neural network every time to add a new class is very time-consuming and inefficient. Some essential adjustments will be needed to implement an incremental learning algorithm using a neural network.

Hyperdimensional computing [3], also known as HD computing, is the core technology that the proposed framework was built on. The ease of implementation and processing that HD computing offers, made it an attractive candidate for this research. As further explained in section III-D, HD computing enabled the use of non-optimized neural network in the proposed framework as well as achieving high accuracy rate.

The goal was to build a system that implements incremental learning with a high recognition rate and fast response time. The proposed system has been tested against several
datasets with different conditions and subjects through a series of extensive experiments to demonstrate the attained accuracy and consistency.

The field of HD computing has yet to witness much growth in facial recognition but has seen some significant results in other classifications fields [4]. Some of the examples include European language identification [5], EMG-based Hand Gesture Recognition [6], the use of EEG for seizure detection [7], [8] and using EEG for emotion recognition with physiological signals [9]. Most importantly, HD computing was used on an ultra low power platform [10].

The contributions of this paper are as follows:
1- The use of HD computing in a facial recognition system with a high count of features.
2- Development of a framework capable of incrementally learning with fast response time.
3- Testing the proposed framework against several open-source face datasets and comparing the results with results from previously published papers.

The following sections in this paper are organized as follows: Section II gives some examples of incremental learning methods in face recognition in the literature. Section III explains the pre-processing stage of the proposed framework, introduces HD computing with a brief background and then discusses the incremental learning algorithm of the proposed framework. Section IV has the results of the experiments and commentary, and Section V concludes the paper.

II. RELATED WORK
In this section, we present a group of approaches that have been implemented in previously published works used in incremental face recognition. These approaches can be mainly categorized in 4 main categories: Unsupervised, Semi-Supervised, Supervised and Hybrid.

A. UNSUPERVISED
In this category, the labels for the classes being learned in both training and testing are absent. The methods under this category are required to identify the patterns associated with the classes in the dataset. Some of the approaches under these categories are:

1) PCA VARIATIONS
Principal Component Analysis (PCA) is a classic dimensionality reduction method used for unsupervised learning in machine learning algorithms to remove noise by reducing the number of features in the system. A variation of PCA can be seen in use in incremental learning in the work of Nakouri and Limam [11], which introduces a QR decomposition based 2DPCA (2-dimensional approach to using PCA) with an SVD decomposition to generate the subspace. This is a computationally efficient algorithm with a disadvantage of accuracy dropping with face variations or noise. Another work that uses a variation PCA is Liang et al. [12], where a deep learning network for online unsupervised feature extraction is implemented with the name (IOCANet). IOCANet consists of two stages of Incremental Orthogonal Component Analysis that generate convolution kernels. Further layers can be added to extract higher level features. The last stage is a layer performing both hashing and histogram processes to extract the features in an unsupervised manner. The advantages of this method are its high efficiency and success rate. The main disadvantage is the computational complexity of the method itself and the low capacity of processing only one image at a time.

2) SVD VARIATIONS
Singular value decomposition (SVD) is another classic dimensionality reduction method. A variation of SVD can be seen in the work of Azary and Savakis [13]. In this work, an incremental version of SVD is used over a Grassmann manifold to embed the faces of the subjects. This approach is very computationally efficient due to the use of QR decomposition and can capture variations of pose, illumination, and expression on a single subject over a group of images. The disadvantage of this approach lies in the deviation that occurs when the manifold of the data differs.

3) OTHER VARIATIONS
Other unsupervised learning approaches to incremental learning methods that don’t strictly fall under PCA or SVD variations include the work of Ye and Yang [14]. The work proposed an Incremental Sparse Representation Classification (SRC) method which divides the face under classification into several parts. The parts that show great variations due to illumination or occlusion are discarded and the rest are used to reconstruct the face that is used for the result. The new data that are used for training in this method only affect the faces’ dictionary for the subjects under training while retaining the rest of the dictionary. The advantage of this method lies in its Divide-and-Rule approach giving it higher efficiency when updating its dictionary while having the disadvantage of throwing away the areas under high variations which may result in the loss of several features. Another work Nakouri and Limam [15] has an incremental version of Generalized Low Rank Approximation of Matrices. The work proposed a method that uses an iterative approach to update the generated low rank structures to form an incremental learning method. The disadvantage of this approach lies in its higher computational complexity.

B. SEMI SUPERVISED
In this category, a small subset of the training data includes labels for the classes. Usually the labeled data is added online while the unlabeled data is a fixed amount of offline data. One of the approaches in this category is the work of Dhamecha et al. [16] that proposed an incremental semi-supervised discriminant analysis method which uses the unlabeled data to enable incremental learning. The initial learning stage uses a big portion of the unlabeled data samples and few labeled data samples to approximate the Total Scatter Matrix. The Between Scatter Matrix is updated with each new
data sample. While this is a very efficient approach for online usage, it is also extremely computationally complex.

C. SUPERVISED
In this category, all the training and testing data are labeled. Some of the approaches under these categories are:

1) LDA VARIATIONS
Linear discriminant analysis (LDA) is a method used to find a linear combination of features that characterizes a set of classes. A variation of LDA can be seen in the work of Soula et al. [17]. In this work, a technique based on Incremental Nonparametric Discriminant Analysis (NDA) is proposed for incremental face recognition. The technique uses Gabor Ordinal Measures as preprocessing while the incremental update algorithm involves the re-computation of the matrices of NDA. This approach has a high efficiency in feature extraction but has the disadvantage of being computationally complex.

2) SVM VARIATIONS
Support Vector Machines (SVM) are a popular supervised learning approach with the added benefit of being able to maximize the gap between the classes. A variation of SVM can be seen in the work of Sisodia et al. [18] which introduced an Incremental version of SVM. This approach uses Discrete Cosine Transformation (DCT) to reduce space complexity in large datasets. The advantage of this approach lies in its low computational complexity. The main disadvantage of it is the approach’s weakness to complex face variations.

3) OTHER VARIATIONS
Other supervised learning approaches to incremental learning methods that don’t strictly fall under LDA or SVM variations include the work of Shi et al. [19]. This work proposed a Min–Max objective to a layer below the output layer of a CNN model while training and uses the gradients and errors to optimize the output of the network. This approach has high accuracy but is impaired by its computational complexity.

D. HYBRID
In the hybrid category, the approaches don’t fall under any specific subset of previously mentioned categories. The work of Buhuş et al. [20] proposed an approach based on the fusion of Local Binary Patterns (LBP) and (DCT) for feature extraction with a Simplified Fuzzy Adaptive Resonance (SFAM) classifier. This approach has high accuracy but is devalued from computational complexity. Another approach in the work of Dinkova et al. [21] which proposed a fusion of SVD with Hidden Markov Model (HMM). This approach divides the face into its basic features (hair, forehead, eyes, etc.) to achieve high accuracy.

With this analysis in mind of the categories of incremental learning, the proposed framework is subsumed in the Hybrid category as a pre-trained neural network “inceptionV3” was used as a main feature extractor with a k-means clustering algorithm to reduce the dimensions of the feature vector. HD computing was used as a classifier in a supervised manner. This approach gives the proposed framework the advantage of being adaptive to face variations due to the pre-processing stage while also being computationally simple with the online stage of incremental learning as well as resistance to noise [22] due to the nature of HD computing as further explained in section III-D.

III. HYPERDIMENSIONAL COMPUTING PARADIGM FOR FACE RECOGNITION

A. INTRODUCTION
In recent years, the advancement in artificial intelligence applications has been steadily getting faster and pushing the boundaries of technology even further. This acceleration can be witnessed due to the introduction of multiple techniques and technologies and combining them to get improved results. The aim of this research is to introduce one such technique by using HD computing as the central part of an incremental learning facial recognition framework. It is believed that with this proposed framework, researchers in the future will have one more usable option to achieve higher success rates with their target projects.

B. ARCHITECTURE OF THE PROPOSED FRAMEWORK
The architecture of the proposed framework is shown in figure 1. The framework consists of 2 blocks: Feature extraction block and HD classifier block. The Feature extraction block envelopes two stages: Feature extraction stage using “inceptionV3” and Features reduction stage using K-means clustering. The feature extraction block is responsible for mapping the input image from the input space into the feature space $\mu$. The process of preparing this stage for usage is explained in section III-C. The HD classifier block uses a combination of an item memory (IM) and an associative memory (AM) to classify the target dataset. An encoding and decoding structure is set before and after the HD classifier block for interaction with the feature extraction block to map the features from the feature space $\mu$ to HD computing space $\tau$. The mapping between the two spaces is done through an encoding from feature space to HD computing structure. The output is extracted using a decoding from HD computing to flat output space structure. The encoding and decoding structure is further explained in section III-D2 while the process of using HD computing in the proposed framework is explained in section III-D4.

C. PRE-PROCESSING
In this section, it is explained how the data was prepared for use by the latter steps along with the explanation of how the first step of training is going to be executed. A pseudo code of the pre-processing algorithm is shown in algorithm figure 1. The pre-processing stage began with getting an image from the target dataset. The image is passed to a neural network. The neural network chosen for the proposed
framework was “inceptionV3” which is a convolutional neural network that is widely used for assisting in image analysis and object detection. No components of “inceptionV3” were re-trained or modified and only the output of one of its layers (max_pooling2d_2) used as output without any alterations. The reasoning behind that was being able to build an easily upgradeable and adjustable framework in which “inceptionV3” was used as a tool for feature extraction without any further optimization for the datasets used. This framework should be able to easily handle a change of the neural network used down the line and be adaptable if it was decided that there was need to completely change the method of feature extraction. The output of the layer chosen was a vector of size (35*35*192).

The next stage of the pre-processing is clustering using K-means to form a bag of visual words. The formation of the used bag of words needed a large number of feature vectors that were extracted earlier from “inceptionV3.” This was done by collecting the output vectors from “inceptionV3” from a considerable number of input images coming from the target dataset. These vectors underwent the process of clustering using K-means clustering of 1000 clusters. After the clustering process was over, the bag of words could be used by presenting a new vector coming from an image passed to “inceptionV3” and the bag of words would output the clusters that contain the image vector signifying the features detected from the image. The number of clusters was chosen through an analysis process to find the number that gives the highest accuracy. Increasing the number of clusters beyond 1000 clusters did not show much increase in accuracy in all the datasets used in the conducted experiments and simply increased the load of usage on the memory during the clustering process.

D. HD COMPUTING

In this section, some background knowledge about HD computing is established, along with its advantages and how it is usually used to better explain the proposed framework. It is also explained why HD computing was chosen as the framework’s incremental learning mechanism how HD computing was used to in the implemented classifier.

1) BACKGROUND

The brain is an extraordinarily complex computational system that has inspired multiple models across many fields.

Algorithm 1 Setting up Bag of Words for Each Dataset.

Input: training images from dataset
Output: Bag of Words out

Initialization:
1: Generate 1000 random binary vectors of size 10000 bits (IM)
2: Generate X empty vectors of size 10000 bits (AM) where X is number of subjects in dataset

LOOP Process
3: for $i = 1$ to numberoftrainingimages do
4: FeatureVector ← inceptionV3(image)
5: Update (FeatureVectorArray) with (FeatureVector)
6: end for

Bag of words formation
7: set K-means clustering to cluster for 1000 means
8: Use (FeatureVectorArray) as input for k-means
9: return BagOfWords

Hyperdimensional (HD) computing is a mathematical computing model inspired by brain circuitry and its sheer size. The HD computing mathematical model is based on operations of binary strings of extremely high dimensionality (e.g., D=10000) that are called HD vectors [3], [23]. Due to their high dimensionality, there are many vectors that are nearly orthogonal in the high dimensional space that is used [3], [24]. There are several well-defined arithmetic operations that can be applied to combine any two or more HD vectors together to form a new one. The resulting HD vector preserves most information from all the combined vectors. These operations include addition, multiplication, and thresholding [4].

The arithmetic operations performed on HD computing are described as follows: a) Addition: also known as bundling, it is the most basic HD vector operation that is commonly used as bit-wise addition operation between the vectors of interest. For example, HD vector X is the result of the bit-wise addition of HD vectors A and B

$$X = A + B$$

$$A = 11 - 1 - 111$$

$$B = 1 - 11 - 11 - 1$$

$$X = 200 - 210$$

From this example, the result is not binary which require a threshold operation which is usually the case after an addition (bundling) operation. b) Multiplication: also known as binding, is a bit-wise multiplication between HD vectors operation which results in an XOR output HD vector. For example, HD vector X is the result of the bit-wise multiplication of HD vectors A and B

$$X = A \oplus B$$

$$A = 11 - 1 - 111$$

$$B = 1 - 11 - 11 - 1$$

$$X = 1 - 1 - 111 - 1$$

$$x = 1 - 1 - 111 - 1$$
c) Thresholding: also known as majority rule, is usually used after addition to determine the resulting HD vector by putting it back into the binary form. To use the majority rule, a boundary should be set. For example, for HD vector Y resulting from thresholding on HD vector X, a rule of any value equal to or greater than 0 is to be set to 1, and any other value is to be set to -1.

\[ X = 200 - 210 \]
\[ Y = \begin{cases} X_i \geq 0 & \rightarrow Y_i = 1 \\ X_i < 0 & \rightarrow Y_i = -1 \end{cases} \]
\[ Y = 111 - 111 \quad (3) \]

To start representing items of interest using HD vectors, one can begin by generating random vectors with equal distribution of +1s and -1s (Binary vectors) on the high dimensional scale (e.g., \(D=10000\)). These HD vectors can then be regarded as the labels used to define an object of interest. To make a classifier using HD computing, a set of random equally distributed vectors should be generated to form an item memory (IM): an array of labels, each defining a label/feature of the object of interest. Vectors from IM are then combined through the operations mentioned previously (addition - multiplication - thresholding). After finishing all operations, the resulting HD vectors are then stored in an associative memory (AM): an array of vectors defining the most significant features of the subject class. The process of filling the AM is referred to as the learning process. During the testing process, the resulting HD vectors are compared to the vectors in the AM using different similarity measures such as Hamming distance or Cosine similarity. For two HD vectors \(X_1\) and \(X_2\) of dimension \(D\), both, Hamming distance is calculated as:

\[ \text{Hamming}(X_1, X_2) = \frac{1}{D} \sum_{i=1}^{D} 1_{X_1(i) \neq X_2(i)} \quad (4) \]

while Cosine similarity is calculated as:

\[ \text{Cosine}(X_1, X_2) = \frac{X_1 \cdot X_2}{|X_1||X_2|} \quad (5) \]

Both calculating methods can be used for binary hyper vectors. For the proposed framework, Cosine similarity was selected since it yields a stronger indicator when two HD vectors have multiple bits of adjacent positions and equal value so it was better for this implementation.

2) ENCODING AND DECODING

For the HD computing classifier to work with the feature extraction block and give a proper output, an encoding and decoding structure was set in place. For a given set of features \((\tau)\) extracted from a data sample, to transform \((\tau)\) to HD space \((\mu)\), Let \((\theta)\): \((\tau) \rightarrow (\mu)\) be a bundling of features. For a data point \(X = \{(x_i) \in (\tau)\}\) for \(i = 1 \rightarrow n\) consists of \(n\) features. The first step to this encoding is through the operation of bundling:

\[ \upsilon(X) = \sum_{i=1}^{n} \theta(x_i) \quad (6) \]

After the bundling operation, a majority rule must be set in place to conclude the encoding to HD space \((\mu)\). For a binary vectors of high dimension \(D\), the majority rule is set with reference to the binary set accepted values \((V_1, V_2)\) and the dimension of the vector \(D\). For case of binary set values \((V_1 = -1, V_2 = +1)\), the majority rule is set with majority threshold being \(D/2\). However, For case The majority rule operation for \((V_1 = -1, V_2 = +1)\) is set as:

\[ \phi = \begin{cases} \upsilon(X) \geq 0 & \rightarrow Y_i = 1 \\ \upsilon(X) < 0 & \rightarrow Y_i = -1 \end{cases} \quad (7) \]

The decoding process starts with the usage of Cosine similarity shown in equation 5. For number of classes \(J\), the Cosine similarity operation is performed on all class vectors \((\psi)\) against encoded vector \((\phi)\). The decoding process concludes with accepting the class vector \((\psi_i)\) that yielded the highest similarity with encoded vector \((\phi)\) as output Y:

\[ Y = \arg\max_{\psi_i} (\text{Cosine}(\psi_i, \phi)) \quad (8) \]

3) INCREMENTAL LEARNING MECHANISM

An incremental learning approach using HD computing to continuously extend the number of classes within the target system in an online manner was implemented. In neural networks, adding a subject for classification after initial training required a change in the structure of the neural network to accommodate the new number of classes in the system, as well as re-training the network. The advantage that HD computing added to the proposed framework is that any addition of subjects to the system does not require any change of structure or re-training. All that was needed was to add a new vector in the AM for the added subject as well as performing the training process for this specific vector only. This was possible due to the independence of the vectors of the subjects in HD computing. This is further explained in the next section.

4) IMPLEMENTATION OF HD COMPUTING

In this section, it is explained how HD computing was used in the proposed framework following the steps of pre-processing mentioned previously in section III-C. A pseudo code of the training algorithm is shown in algorithm 2. After the bag of words was ready to use at the end of the pre-processing stage, the AM formation was started by generating random HD vectors that became the IM. Each of these vectors was a unique equally distributed binary 10000-bit vector. In this framework, the bits could only be of value \(+1\) or \(-1\). The number of vectors in IM is equal to the number of clusters inside the bag of words. In this subject case, 1000 clusters were chosen through through experimentation to find the optimal range for highest success rate across
all of the used datasets in the performed tests. An empty AM was generated to host the subject specific vectors at the end of the training process. The number of vectors in the AM was the number of subject classes to be classified in the target dataset. The training and testing process implement the encoding and decoding structure previously shown in section III-D2.

The training process of each class is the same. A pseudo code of the testing algorithm is shown in algorithm 3. The initial steps are the same as the pre-processing stage. An image of the target subject is passed to “inceptionV3” [25] to extract the feature vector. The extracted feature vector is passed to the bag of words and an array of integers was returned as an output. The output of the bag of words corresponded to which features were extracted from the image. After that, the vectors from IM that correspond to the array of integers coming out of the bag of words were collected. The vectors collected from IM were then added to each other in a bit-wise manner and additionally added to the vector of the subject in the AM in a bit-wise manner. This process is repeated for each image for the same class. After all the images were processed, a thresholding operation was applied to all the classes vectors in the AM as mentioned in section III-D1.

The testing process is similar to the training process. An image that was subject to classification was passed to “inceptionV3” to extract the feature vector. The extracted feature vector is passed to the bag of words, and an array of integers was returned as an output. Then the vectors from IM that correspond to the array of integers coming out of the bag of words were taken and added to each other in a bit-wise manner. A thresholding operation is applied on the resulting vector and thereby getting the testing vector. To determine which class the tested image belongs to, the testing vector was tested against all the vectors in the AM using the cosine similarity. To add any more subjects to the system after the training process is done, all that was required was to add an empty vector to the AM and apply the same training process to the images of the new class. This process does not affect any of the previous vectors in any way resulting in an easy to scale incremental learning approach.

IV. EXPERIMENTAL RESULTS

In this section, the fundamental properties and performance of the proposed framework is tested on some open-source face datasets. To measure the proposed framework’s performance in different conditions, 11 datasets were used (CBCL [26], NCKU [27], [28], Color Ferret [29], [30], MUCT [31], CMU [32], [33], UMIST [34], [35], Yale2B [36], [37], face94 [38], grimace [39], ORL/Olivette [40], [41], AR [42], [43]). Information about the data sets can be found in table 1. The normal manner of implementing a machine learning algorithm into a classifying system requires training the algorithm on classifying a set of subjects given some training samples. In the tests conducted, the first phase involved the pre-processing step and followed by the training and testing of several classes. The second phase incorporated adding more subjects after the first phase ended and testing the recognition rate of those classes.

```plaintext
Algorithm 2 HD Training.
Input: training images from dataset
Output: AM out
Initialization:
1: Generate 1000 random binary vectors of size 10000 bits (IM)
2: Generate X empty vectors of size 10000 bits (AM) where X is number of subjects in dataset
LOOP Process
3: for i = 1 to numberoftrainingimages do
4: FeatureVector ← inceptionV3(image)
5: OutputWords ← BagOfWords(FeatureVector)
6: AM(OutputWords) ← AM(OutputWords) + IM(OutputWords)
end for
Thresholding Process
8: for All Vectors in AM do
9: for All Bits in Vector do
10: if (Bit > 0) then
11: 1
12: else
13: -1
14: end if
15: end for
end for
17: return AM
```

```plaintext
Algorithm 3 HD Testing.
Input: image to be tested from dataset
Output: Answer out
Initialization
1: FeatureVector ← inceptionV3(image)
2: OutputWords ← BagOfWords(FeatureVector)
3: OutputVector ← IM(OutputWords)
Thresholding Process
4: for All Bits in OutputVector do
5: if (Bit > 0) then
6: 1
7: else
8: -1
9: end if
10: end for
Output
11: for i = 1 to numberofsubjectsinAM do
12: SimilarityArray(i) ← cosineSimilarity(OutputVector, AM(i) )
13: end for
14: Answer ← argmax (SimilarityArray)
15: return Answer
```

Initially, the recognition rate of the proposed framework was verified on all the datasets in the first phase. Next, the recognition rate on the second phase was checked. After that, the incremental learning mechanism’s degradation of recognition rate was analyzed by continuously adding more subjects in an incremental fashion repeating the second phase multiple times. These tests were run on a PC running Windows 10 64-bit on an Intel Core i5-8600k CPU and 16 GB of memory using MATLAB 2020a. The tests did not include any use of code running on a GPU.

### A. TESTING FIRST AND SECOND PHASE RECOGNITION RATE

The process started off by testing the accuracy of the proposed framework in the first phase by using 70% of the dataset and saving the rest for the second phase. In both first phase and second phase, images of each subject were split into 75% for training and 25% for testing. The 25% of the images saved for testing were never used in the training phase. The 3:1 training to testing ratio was applied in both phases. This test was run on all the datasets and applied 10 times to ensure consistent performance across all trials. The success rate for the testing in both first phase and second phase were recorded and can be seen in table 2. The success rate is extremely high on most of the datasets used except (FERET, UCI, AR). This can be attributed to the occlusion on some of the images in AR and UCI with subjects wearing scarves or sunglasses. As for FERET, some of the subjects have more images than others with subjects with more images have images with different angles and backgrounds as shown in figure 2. The time elapsed for each image in both training and testing has been recorded and can be seen in table 3. The processing time for each image in both phases is extremely low due to the low computational complexity of the proposed framework. In FERET, the testing takes more time than training per image due to the enormous number of test subjects in comparison to the other datasets. A comparison between the results of the proposed framework and the results of previously published incremental learning systems can be seen in table 4. While the proposed framework does not have the highest success rate at each dataset, it shows a very consistent recognition rate in comparison to the other techniques. This can be attributed to the ability of the proposed framework to generalize more than the other techniques. This confirms the proposed framework’s goal in taking the first step of using HD

| dataset       | First Phase (%) | Second Phase (%) |
|---------------|----------------|------------------|
| Yale2B        | 99.6           | 99.44            |
| NCKU          | 99.04          | 97.9784          |
| Feret         | 85.14          | 78.84            |
| MUCT          | 99.72          | 98.91            |
| UCI           | 91.74          | 80.33            |
| UMIST         | 99.69          | 98.35            |
| CBCL          | 97.84          | 95.196           |
| face94        | 99.61          | 99.299           |
| Grimace       | 100            | 100              |
| ORL/Olivetti  | 99.13          | 98.78            |
| AR            | 95.18          | 90.998           |

| dataset       | Training (msec) | Testing (msec) |
|---------------|----------------|----------------|
| Yale2B        | 122            | 102            |
| NCKU          | 114            | 90             |
| Feret         | 144            | 193            |
| MUCT          | 115            | 103            |
| UCI           | 115            | 97             |
| UMIST         | 123            | 101            |
| CBCL          | 121            | 101            |
| face94        | 136            | 101            |
| Grimace       | 119            | 94             |
| ORL/Olivetti  | 133            | 102            |
| AR            | 118            | 98             |

**TABLE 1.** Info about all the data sets used in this paper.

| Data set       | No. subjects | No. images |
|----------------|--------------|------------|
| CBCL           | 9            | 1859       |
| NCKU           | 90           | 6660       |
| FERET          | 994          | 11338      |
| MUCT           | 276          | 3755       |
| CMU            | 20           | 1872       |
| UNIMIST        | 20           | 1012       |
| Yale2B         | 28           | 16380      |
| face94         | 153          | 3078       |
| ORL/Olivetti   | 40           | 400        |
| AR             | 136          | 3315       |

| Info about the data set                                                                 |
|-------------------------------------------------------------------------------------------|
| The subjects were asked to rotate their faces in depth and the lighting conditions were changed by moving a light source around the subject  |
| images were taken every 5 degrees from right profile to left profile in the pan rotation  |
| Many subjects with different poses and facial expressions                                  |
| diversity of lighting, age, and ethnicity                                                |
| face images of people taken with varying pose (straight, left, right, up), expression (neutral, happy, sad, angry), eyes (wearing sunglasses or not), and size |
| Each subject is shown in a range of poses from profile to frontal views                  |
| Subjects were speaking while a sequence of twenty images was taken to introduce moderate and natural facial expression variation |
| A sequence of 20 images per individual was taken, using a fixed camera. During the sequence, the subject moves his/her head and makes grimaces which get more extreme towards the end of the sequence |
| Images taken at various times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses) |
| Different facial expressions, illumination conditions and occlusions                      |

**TABLE 2.** The average recognition rate results of the proposed framework in percentage.

**TABLE 3.** The average processing time of a single image during the testing of the proposed framework in milliseconds.
TABLE 4. The results of different incremental facial recognition techniques against the proposed framework.

| dataset     | DSLDA (2009) [46] | I2DKPCA (2013) [47] | 2DIPCA (2016) [11] | OLDAS (2016) [48] | ISLRC (2018) [49] | INDA (2018) [17] | Incremental PCA (2006) [50] | FFSF (2021) [51] | Ours  |
|-------------|-------------------|---------------------|-------------------|------------------|------------------|------------------|-----------------------|------------------|-------|
| Yale2B      | 89.30%            | 83%                 | 91.40%            | 97.94%           | 89.90%           | 96%              | 99.13                 | 99.36%           |       |
| Ferret      |                   |                     |                   |                  |                  |                  |                       |                  | 79.53% |
| ORL         |                   |                     |                   |                  |                  |                  |                       |                  | 97.41% |
| AR          | 96.53%            | 82.92%              | 83%               |                  |                  |                  |                       |                  | 91.91% |

FIGURE 2. data sets (a)MUCT and (b)NCKU having higher recognitions rate can be associated with consistent number of images per subject with same image conditions while (c)AR and (d)UCI have some images that include occlusion due to the use of sunglasses and scarves. In (e)FERRET we can see different number of images per subject is included as well as changing the type of image.

FIGURE 3. the results of testing the proposed framework under different conditions for first phase.

FIGURE 4. the results of testing the proposed framework under different conditions for second phase.

B. TESTING ROBUSTNESS OF THE PROPOSED FRAMEWORK UNDER DIFFERENT INITIAL CONDITIONS

To test the robustness of the proposed framework under different initial conditions, the percentages of the first phase and second phase subjects were changed. Keeping the same conditions of the training to testing ratio as the previous test. The same test was executed repeatedly by only changing the percentages of the datasets given at each phase by moving 10% of the dataset subjects from the first phase to the second phase each run. The results can be seen in figures 3 and 4. The results proved that the proposed framework is robust enough to maintain its recognition rate with any changes to the initial conditions.

C. TESTING DEGRADATION OF RECOGNITION RATE WITH CONTINUOUS INCREMENTS

The final test was conducted to evaluate the incremental learning mechanism of the proposed framework with respect to the degradation of recognition rate. The datasets used for this test are (NCKU, FERET, MUCT, Face94, AR). The reason to only use these 5 datasets stems from the number of subjects within each dataset. At least 90 unique subjects per dataset were needed for the test. In this experiment, the first phase started with 40 subjects being used. The number of subjects in the first phase remained constant among all runs of the test. To test the incremental learning mechanism, 5 new subjects were added that were not previously part of the AM (Associative Memory). This process was repeated 10 times for a total of 50 new subjects added to the system after the first phase. After the process of adding 5 new subjects, the accuracy of the proposed framework was assessed to test the degradation of recognition rate. The results are in figure 5. The results indicate a very consistent recognition rate across all the phases.
This paper introduced the use of HD computing within a framework for facial recognition. The proposed framework combines the use of pre-trained neural networks as well as unsupervised clustering. Facial recognition is one of the areas that has not witnessed extensive use of HD computing. After-mentioned framework was able to achieve a high recognition rate as well as time efficient training and testing. Through the experimental tests, the scalability of this framework was proven by using it on various sizes of data sets. The tests also proved the robustness of the framework dealing with different data sets and initial conditions.

For future work, The focus will be on the upgradeability of the proposed framework. Taking into consideration the framework’s weakness in dealing with different backgrounds and facial occlusion specifically, different approaches for the feature extraction mechanism will be in consideration. Last but not least there are also plans to optimize the system for large scale data sets with focus on time efficiency. In conclusion, there are many more avenues of research that can stem from this framework and its implementations, which can help facilitate the future of AI and forward the field.

ACKNOWLEDGMENT

The authors would like to thank Sandra Lumine for proofreading the article.

REFERENCES

[1] S. Madhavan and N. Kumar, “Incremental methods in face recognition: A survey,” Artif. Intell. Rev., vol. 54, no. 1, pp. 253–303, Jan. 2021.
[2] M. Masana, X. Liu, B. Twardowski, M. Menta, A. D. Bagdanov, and J. van de Weijer, “Class-incremental learning: Survey and performance evaluation on image classification,” 2020, arXiv:2010.15277.
[3] P. Kanerva, “Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors,” Cognit. Comput., vol. 1, no. 2, pp. 139–159, Oct. 2009.
[4] L. Ge and K. K. Parhi, “Classification using hyperdimensional computing: A review,” IEEE Circuits Syst. Mag., vol. 20, no. 2, pp. 30–47, 2020.
[5] A. Rahimi, P. Kanerva, and J. M. Rabaey, “A robust and energy-efficient classifier using brain-inspired hyperdimensional computing,” in Proc. Int. Symp. Low Power Electron. Design, 2016, pp. 64–69.
[6] A. Moin, A. Zhou, A. Rahimi, S. Benatti, A. Menon, S. Tamakloe, J. Ting, N. Yamamoto, Y. Khan, F. Burghardt, L. Benini, A. C. Arias, and J. M. Rabaey, “An EMG gesture recognition system with flexible high-density sensors and brain-inspired high-dimensional classifier,” in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), May 2018, pp. 1–5.
[7] A. Burrello, K. Schindler, L. Benini, and A. Rahimi, “One-shot learning for iEEG seizure detection using end-to-end binary operations: Local binary patterns with hyperdimensional computing,” in Proc. IEEE Biomed. Circuits Syst. Conf. (BioCAS), Oct. 2018, pp. 1–4.
[8] A. Burrello, L. Cavigelli, K. Schindler, L. Benini, and A. Rahimi, “Lae- lay: An energy-efficient seizure detection algorithm from long-term human iEEG recordings without false alarms,” in Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE), Mar. 2019, pp. 752–757.
[9] E.-J. Chang, A. Rahimi, L. Benini, and A.-Y.-A. Wu, “Hyperdimensional computing-based multimodality emotion recognition with physiological signals,” in Proc. IEEE Int. Conf. Artif. Intell. Circuits Syst. (AICAS), Mar. 2019, pp. 137–141.
[10] F. Montagana, A. Rahimi, S. Benatti, D. Rossi, and L. Benini, “PULP-HD: Accelerating brain-inspired high-dimensional computing on a parallel ultra-low power platform,” in Proc. 55th ACM/ESDA/IEEE Design Autom. Conf. (DAC), Jun. 2018, pp. 1–6.
[11] H. Nakouri and M. Limam, “An incremental two-dimensional principal component analysis for image compression and recognition,” in Proc. 12th Int. Conf. Signal-Image Technol. Internet-Based Syst. (SITIS), 2016, pp. 725–731.
[12] Y. Liang, Y. Yang, F. Shen, J. Zhao, and T. Zhu, “An incremental deep learning network for on-line unsupervised feature extraction,” in Proc. Int. Conf. Neural Inf. Process. Springer, 2017, pp. 383–392. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-70096-0_40
[13] S. Azary and A. Savakis, “Continuous recognition with incremental learning on Grassmann manifolds,” in Proc. IEEE 60th Int. Midwest Sympt. Circuits Syst. (MWSCAS), Aug. 2017, pp. 1477–1480.
[14] J. Ye and R. Yang, “An incremental SRC method for face recognition,” in Proc. Pacific Rim Conf. Multimedia. Springer, 2015, pp. 170–180. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-24078-7_17
[15] H. Nakouri and M. Limam, “Incremental generalized low rank approximation of matrices for visual learning and recognition,” Pattern Recognit. Image Anal., vol. 25, no. 1, pp. 68–72, Jan. 2015.
[16] T. W. Bhumecha, R. Singh, and M. Vatsa, “On incremental semi-supervised discriminant analysis,” Pattern Recognit., vol. 52, pp. 135–147, Apr. 2016.
[17] A. Soulia, S. Ben Said, R. Ksantini, and Z. Lachiri, “A novel incremental face recognition method based on nonparametric discriminant model,” in Proc. 4th Int. Conf. Adv. Technol. Signal Image Process. (ATISP), Mar. 2018, pp. 1–6.
[18] D. Sisodia, L. Singh, and S. Sisodia, “Incremental learning algorithm for face recognition using DCT,” in Proc. IEEE Int. Conf. Emerg. Trends Comput., Commun. Nanotechnol. (ICECCN), Mar. 2013, pp. 282–286.
[19] W. Shi, Y. Gong, X. Tao, J. Wang, and N. Zheng, “Improving CNN performance accuracies with min–max objective,” IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 7, pp. 2872–2885, Jul. 2018.
[20] E. R. Buhus, L. Grama, and C. Serbu, “A facial recognition application based on incremental supervised learning,” in Proc. 13th IEEE Int. Conf. Intell. Comput. Commun. Process. (ICCPP), Sep. 2017, pp. 279–286.
[21] P. Kanerva, P. Georgieva, A. Manolova, and M. Milanova, “Face recognition based on subject dependent hidden Markov models,” in Proc. IEEE Int. Black Sea Conf. Commun. Netw. (BlackSeaCom), Jun. 2016, pp. 1–5.
[22] A. Thomas, S. Dasgupta, and T. Rosing, “Theoretical foundations of hyperdimensional computing,” J. Artif. Intell. Res., vol. 72, pp. 215–249, Oct. 2021.
[23] P. Kanerva, “Computing with 10,000-bit words,” in Proc. 52nd Annua. Allerton Conf. Commun., Control. Comput. (Allerton), Sep. 2014, pp. 304–310.
[24] P. Kanerva, “Binary spatter-coding of ordered k-tuples,” in Proc. Int. Conf. Artif. Neural Inf. Process. Springer, 1996, pp. 869–873. [Online]. Available: https://link.springer.com/chapter/10.1007/3-540-61510-5_146
[25] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in Proc. IEEE Conf. Artif. Neural Netw. (ICANN), Sep. 2017, pp. 304–310.
[26] A. Rahimi, P. Kanerva, and J. M. Rabaey, “A robust and energy-efficient classifier using brain-inspired hyperdimensional computing,” in Proc. IEEE Int. Conf. Face Recognition, 2003, pp. 725–731.
[27] T. H. Wang and J.-J. J. Lien, “Facial expression recognition system based on rigid and non-rigid motion separation and 3D pose estimation,” Pattern Recognit., vol. 42, no. 5, pp. 962–977, 2009.
[28] N. C. K. University. Facedetect and Poseestimate Data Set. Accessed: Sep. 8, 2021. [Online]. Available: http://robotics.cise.neu.edu.tw/Databases/FaceDetect_PoseEstimate.htm
