**AIM:**

Children are the greatest asset of each nation. The future of any country depends upon the right upbringing of its children. India is the second populous country in the world and children represent a significant percentage of total population

**PROBLEM STATEMENT**

But unfortunately a large number of children go missing every year in India due to various reasons including abduction or kidnapping, run-away children, trafficked children and lost children. A deeply disturbing fact about India’s missing children is that while on an average 174 children go missing every day, half of them remain untraced.

**ABSTRACT**

In India a countless number of children are reported missing every year. Among the missing child cases a large percentage of children remain untraced. This paper presents a novel use of deep learning methodology for identifying the reported missing child from the photos of multitude of children available, with the help of face recognition. The public can upload photographs of suspicious child into a common portal with landmarks and remarks. The photo will be automatically compared with the registered photos of the missing child from the repository. Classification of the input child image is performed and photo with best match will be selected from the database of missing children. For this, a deep learning model is trained to correctly identify the missing child from the missing child image database provided, using the facial image uploaded by the public. The Convolutional Neural Network (CNN), a highly effective deep learning technique for image based applications is adopted here for face recognition. Face descriptors are extracted from the images using a pre-trained CNN model VGG-Face deep architecture. Compared with normal deep learning applications, our algorithm uses convolution network only as a high level feature extractor and the child recognition is done by the trained SVM classifier. Choosing the best performing CNN model for face recognition, VGG-Face and proper training of it results in a deep learning model invariant to noise, illumination, contrast, occlusion, image pose and age of the child and it outperforms earlier methods in face recognition based missing child identification. The classification performance achieved for child identification system is 99.41%. It was evaluated on 43 Child cases.

**INTRODUCTION**

Children are the greatest asset of each nation. The future of any country depends upon the right upbringing of its children. India is the second populous country in the world and children represent a significant percentage of total population. But unfortunately a large number of children go missing every year in India due to various reasons including abduction or kidnapping, run-away children, trafficked children and lost children. A deeply disturbing fact about India’s missing children is that while on an average 174 children go missing every day, half of them remain untraced. Children who go missing may be exploited and abused for various purposes. As per the National Crime Records Bureau (NCRB) report which was cited by the Ministry of Home Affairs (MHA) in the Parliament (LS Q no. 3928, 20-03- 2018), more than one lakh children (1,11,569 in actual numbers) were reported to have gone missing till 2016, and 55,625 of them remained untraced till the end of the year. Many NGOs claim that estimates of missing children are much higher than reported. Mostly missing child cases are reported to the police. The child missing from one region may be found in another region or another state, for various reasons. So even if a child is found, it is difficult to identify him/her from the reported missing cases. A framework and methodology for developing an assistive tool for tracing missing child is described in this paper. An idea for maintaining a virtual space is proposed, such that the recent photographs of children given by parents at the time of reporting missing cases is saved in a repository. The public is given provision to voluntarily take photographs of children in suspected situations and uploaded in that portal. Automatic searching of this photo among the missing child case images will be provided in the application. This supports the police officials to locate the child anywhere in India. When a child is found, the photograph at that time is matched against the images uploaded by the Police/guardian at the time of missing. Sometimes the child has been missing for a long time. This age gap reflects in the images since aging affects the shape of the face and texture of the skin. The feature discriminator invariant to aging effects has to be derived. This is the challenge in missing child identification compared to the other face recognition systems. Also facial appearance of child can vary due to changes in pose, orientation, illumination, occlusions, noise in background etc. The image taken by public may not be of good quality, as some of them may be captured from a distance without the knowledge of the child. A deep learning [1] architecture considering all these constrain is designed here. The proposed system is comparatively an easy, inexpensive and reliable method compared to other biometrics like finger print and iris recognition systems.

THE EXISTING SYSTEM

Mostly missing child cases are reported to the police. The child missing from one region may be found in another region or another state, for various reasons. So even if a child is found, it is difficult to identify him/her from the reported missing cases. A framework and methodology for developing an assistive tool for tracing missing child is described in this paper. An idea for maintaining a virtual space is proposed, such that the recent photographs of children given by parents at the time of reporting missing cases is saved in a repository.

**DISADVANTAGES:**

Earliest methods for face recognition commonly used computer vision features such as HOG, LBP, SIFT, or SURF.However, features extracted using a CNN network for getting facial representations gives better performance in face recognition than handcrafted features.

.

**LITERATURE REVIEW**

Earliest methods for face recognition commonly used computer vision features such as HOG, LBP, SIFT, or SURF [2-3]. However, features extracted using a CNN network for getting facial representations gives better performance in face recognition than handcrafted features. In [4], missing child identification is proposed which employees principal component analysis using Eigen vectors is used for face recognition system. FindFace is a website that lets users search for members of the social network VK by uploading a photograph [5]. FindFace employs a facial recognition neural network algorithm developed by N-Tech Lab to match faces in the photographs uploaded by its users against faces in photographs published on VK, with a reported accuracy of 70 percent. The “Tuanyuan”, or “reunion” in Chinese, app developed by Alibaba Group Holding Ltd. helped Chinese authorities recover hundreds of missing children [6]. The app has allowed police officers to share information and work together with public

**IN “Y. LECUN, Y. BENGIO, AND G. HINTON, "DEEP LEARNING", NATURE, 521(7553):436–444, 2015”** Deep learning allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. These methods have dramatically improved the state-of-the-art in speech recognition, visual object recognition, object detection and many other domains such as drug discovery and genomics. Deep learning discovers intricate structure in large data sets by using the backpropagation algorithm to indicate how a machine should change its internal parameters that are used to compute the representation in each layer from the representation in the previous layer. Deep convolutional nets have brought about breakthroughs in processing images, video, speech and audio, whereas recurrent nets have shone light on sequential data such as text and speech. Machine-learning technology powers many aspects of modern society: from web searches to content filtering on social networks to recommendations on e-commerce websites, and it is increasingly present in consumer products such as cameras and smartphones. Machine-learning systems are used to identify objects in images, transcribe speech into text, match news items, posts or products with users’ interests, and select relevant results of search. Increasingly, these applications make use of a class of techniques called deep learning. Conventional machine-learning techniques were limited in their ability to process natural data in their raw form. For decades, constructing a pattern-recognition or machine-learning system required careful engineering and considerable domain expertise to design a feature extractor that transformed the raw data (such as the pixel values of an image) into a suitable internal representation or feature vector from which the learning subsystem, often a classifier, could detect or classify patterns in the input. Representation learning is a set of methods that allows a machine to be fed with raw data and to automatically discover the representations needed for detection or classification. Deep-learning methods are representation-learning methods with multiple levels of representation, obtained by composing simple but non-linear modules that each transform the representation at one level (starting with the raw input) into a representation at a higher, slightly more abstract level. With the composition of enough such transformations, very complex functions can be learned. For classification tasks, higher layers of representation amplify aspects of the input that are important for discrimination and suppress irrelevant variations. An image, for example, comes in the form of an array of pixel values, and the learned features in the first layer of representation typically represent the presence or absence of edges at particular orientations and locations in the image. The second layer typically detects motifs by spotting particular arrangements of edges, regardless of small variations in the edge positions. The third layer may assemble motifs into larger combinations that correspond to parts of familiar objects, and subsequent layers would detect objects as combinations of these parts. The key aspect of deep learning is that these layers of features are not designed by human engineers: they are learned from data using a general-purpose learning procedure. Deep learning is making major advances in solving problems that have resisted the best attempts of the artificial intelligence community for many years. It has turned out to be very good at discovering intricate structures in high-dimensional data and is therefore applicable to many domains of science, business and government. In addition to beating records in image recognition1–4 and speech recognition5–7, it has beaten other machine-learning techniques at predicting the activity of potential drug molecules8 , analysing particle accelerator data9,10, reconstructing brain circuits11, and predicting the effects of mutations in non-coding DNA on gene expression and disease12,13. Perhaps more surprisingly, deep learning has produced extremely promising results for various tasks in natural language understanding14, particularly topic classification, sentiment analysis, question answering15 and language translation16,17. We think that deep learning will have many more successes in the near future because it requires very little engineering by hand, so it can easily take advantage of increases in the amount of available computation and data. New learning algorithms and architectures that are currently being developed for deep neural networks will only accelerate this progress. Supervised learning The most common form of machine learning, deep or not, is supervised learning. Imagine that we want to build a system that can classify images as containing, say, a house, a car, a person or a pet. We first collect a large data set of images of houses, cars, people and pets, each labelled with its category. During training, the machine is shown an image and produces an output in the form of a vector of scores, one for each category. We want the desired category to have the highest score of all categories, but this is unlikely to happen before training. We compute an objective function that measures the error (or distance) between the output scores and the desired pattern of scores. The machine then modifies its internal adjustable parameters to reduce this error. These adjustable parameters, often called weights, are real numbers that can be seen as ‘knobs’ that define the input–output function of the machine. In a typical deep-learning system, there may be hundreds of millions of these adjustable weights, and hundreds of millions of labelled examples with which to train the machine. To properly adjust the weight vector, the learning algorithm computes a gradient vector that, for each weight, indicates by what amount the error would increase or decrease if the weight were increased by a tiny amount. The weight vector is then adjusted in the opposite direction to the gradient vector. The objective function, averaged over all the training examples, can be seen as a kind of hilly landscape in the high-dimensional space of weight values. The negative gradient vector indicates the direction of steepest descent in this landscape, taking it closer to a minimum, where the output error is low on average. In practice, most practitioners use a procedure called stochastic gradient descent (SGD). This consists of showing the input vector for a few examples, computing the outputs and the errors, computing the average gradient for those examples, and adjusting the weights accordingly. The process is repeated for many small sets of examples from the training set until the average of the objective function stops decreasing. It is called stochastic because each small set of examples gives a noisy estimate of the average gradient over all examples. This simple procedure usually finds a good set of weights surprisingly quickly when compared with far more elaborate optimization techniques18. After training, the performance of the system is measured on a different set of examples called a test set. This serves to test the generalization ability of the machine — its ability to produce sensible answers on new inputs that it has never seen during training. Many of the current practical applications of machine learning use linear classifiers on top of hand-engineered features. A two-class linear classifier computes a weighted sum of the feature vector components. If the weighted sum is above a threshold, the input is classified as belonging to a particular category. Since the 1960s we have known that linear classifiers can only carve their input space into very simple regions, namely half-spaces separated by a hyperplane19. But problems such as image and speech recognition require the input–output function to be insensitive to irrelevant variations of the input, such as variations in position, orientation or illumination of an object, or variations in the pitch or accent of speech, while being very sensitive to particular minute variations (for example, the difference between a white wolf and a breed of wolf-like white dog called a Samoyed). At the pixel level, images of two Samoyeds in different poses and in different environments may be very different from each other, whereas two images of a Samoyed and a wolf in the same position and on similar backgrounds may be very similar to each other. A linear classifier, or any other ‘shallow’ classifier operating on

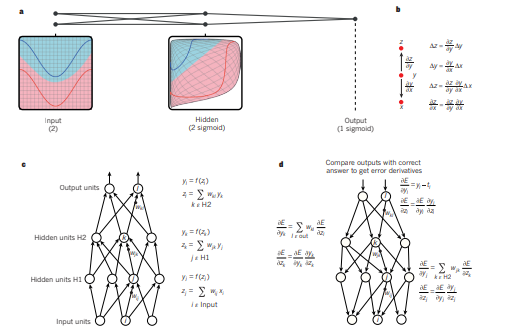


Figure 1 | Multilayer neural networks and backpropagation

**IN “O. DENIZ, G. BUENO, J. SALIDO, AND F. D. LA TORRE, "FACE RECOGNITION USING HISTOGRAMS OF ORIENTED GRADIENTS", PATTERN RECOGNITION LETTERS, 32(12):1598–1603, 2011.”** Face recognition has been a long standing problem in computer vision. Recently, Histograms of Oriented Gradients (HOGs) have proven to be an effective descriptor for object recognition in general and face recognition in particular. In this paper, we investigate a simple but powerful approach to make robust use of HOG features for face recognition. The three main contributions of this work are: First, in order to compensate for errors in facial feature detection due to occlusions, pose and illumination changes, we propose to extract HOG descriptors from a regular grid. Second, fusion of HOG descriptors at different scales allows to capture important structure for face recognition. Third, we identify the necessity of performing dimensionality reduction to remove noise and make the classification process less prone to overfitting. This is particularly important if HOG features are extracted from overlapping cells. Finally, experimental results on four databases illustrate the benefits of our approach ision. It has recently attracted significant attention due to the accessibility of inexpensive digital cameras and computers, and its applications in biometrics and surveillance (see Zhao et al. (2003); Chellappa et al. (1995); Samal and Iyengar (1992); Chellappa and Zhao (2005) for recent surveys of face recognition). Central to the success of face recognition are the feature representation and the classification method. In this paper, we will focus on the former. Broadly speaking, we could classify the features for face recognition as geometric or photometric (view based). The latter seem to have prevailed in the literature (Zhao et al., 2003). There exist a large number of features, starting from the influential Eigenfaces (Principal Component Analysis) (Turk and Pentland, 1991), Gabor wavelets (Amin and Yan, 2009), Local Binary Patterns (Ahonen et al., 2004), Error-Correcting Output Codes (Kittler et al., 2001) and Independent Component Analysis (ICA) (Bartlett et al., 2002) among others. Histograms of Oriented Gradients (HOGs) (Lowe, 2004) are image descriptors invariant to 2D rotation which have been used in many different problems in computer vision, such as pedestrian detection (Bertozzi et al., 2007; Wang and Lien, 2007; Chuang et al., 2008; Watanabe et al., 2009; Baranda et al., 2008; He et al., 2008; Kobayashi et al., 2008; Suard et al., 2006; Zhu et al., 2006; Perdersoli et al., 2007a,b). Recently, in (Albiol et al., 2008) the authors successfully applied HOG descriptors to the problem of face recognition. In that work, a set of 25 facial landmarks were first localized using the Elastic Bunch Graph Matching framework (see Wiskott et al., 1997). The HOG features extracted from the vicinity in each of these 25 facial landmarks were used for classification, using nearest neighbor and Euclidean distance. In this paper, following (Albiol et al., 2008), we further explore the representational power of HOG features for face recognition, and propose a simple but powerful approach to build robust HOG descriptors. In particular, three are the main novelties: (1) build the HOG descriptor using a regular grid, (2) build and combine the HOG descriptors at different scales, (3) apply a linear dimensionality reduction to remove noise, make the classifier more efficient (i.e. reduce dimensionality) and less prone to overfitting. Our results in four standard face databases support the proposed method. The algorithm for extracting HOGs (see Dalal and Triggs, 2005; Lowe, 2004) counts occurrences of edge orientations in a local neighborhood of an image. In our case, the image is first divided into small connected regions, called cells, and for each cell a histogram of edge orientations is computed. The histogram channels are evenly spread over 0–180 or 0–360, depending on whether the gradient is ‘unsigned’ or ‘signed’. The histogram counts are normalized to compensate for illumination. This can be done by accumulating a measure of local histogram energy over the somewhat larger connected regions and using the results to normalize all cells in the block. The combination of these histograms represents the final HOG descriptor. Invariance to scale and rotation may be also achieved by extracting descriptors from only salient points (keypoints) in the scale space of the image following a rotation normalization. The steps involved are: (1) Scale-space extrema detection. (2) Orientation assignment. (3) Descriptor extraction. The first step is intended to achieve scale invariance. The second step finds the dominant gradient orientation. All the orientation counts are then made relative to this dominant direction. Fig. 1 shows an example patch with their corresponding HOGs. In (Albiol et al., 2008) (and the shorter version Monzo et al., 2008) the authors successfully applied HOG descriptors to the problem of face recognition. In that work, faces were previously normalized in scale and orientation, so the steps for scale-space extrema detection and orientation were not necessary. A set of 25 facial landmarks were localized using the Elastic Bunch Graph Matching framework (see Wiskott et al., 1997) with HOG features. The HOG features extracted from the vicinity of each of these 25 facial landmarks were used for classification, using nearest neighbor and Euclidean distance. It is important to note that for a new testing face, the matching procedure makes use of the eye’s position (the eyes were assumed to be at a fixed position after normalization). A potential drawback of the approach taken in (Albiol et al., 2008) is that the final error may crucially depend on the reliability of the landmark localizations. Our hypothesis in this paper is that such approach may not work well when landmarks are not precisely localized due to occlusions, strong illuminations or pose changes. Thus, in this work we propose to first normalize the face and then extract HOG features from a regular grid. The grid is formed by placing equal side patches around a first cell centered in the image, until the whole image is covered. On the other hand, the size of the patch used to extract the HOG features is important. In (Albiol et al., 2008) the best size for the patch was estimated via cross-validation in the Yale database, prior to using the FERET database for the final experiments. The locality of the extracted features is determined by the patch size. We hypothesize that a better result could be obtained by combining information from different patch sizes. The fusion strategy that will be considered here is the product combination of the classifiers at patch sizes. Note that this combination rule is not optimal since it assumes independence of the classifiers, though empirically has performed well in our experiments

**IN “C. GENG AND X. JIANG, "FACE RECOGNITION USING SIFT FEATURES", IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING(ICIP), 2009”** Scale Invariant Feature Transform (SIFT) has shown to be a powerful technique for general object recognition/detection. In this paper, we propose two new approaches: VolumeSIFT (VSIFT) and Partial-Descriptor-SIFT (PDSIFT) for face recognition based on the original SIFT algorithm. We compare holistic approaches: Fisherface (FLDA), the null space approach (NLDA) and Eigenfeature Regularization and Extraction (ERE) with feature based approaches: SIFT and PDSIFT. Experiments on the ORL and AR databases show that the performance of PDSIFT is significantly better than the original SIFT approach. Moreover, PDSIFT can achieve comparable performance as the most successful holistic approach ERE and significantly outperforms FLDA and NLDA. The ability to recognize human faces is a demonstration of incredible human intelligence. Psychologists concluded that holistic and feature based approaches are dual routes to the face recognition [1]. Most early approaches in face recognition extract local features from face images. However, the type of local features which are most stable and discriminative for face recognition is unknown. Due to difficulties in robustly extracting local features from face images, researchers began to use the whole face region as the raw input to a recognition system, and developed holistic matching methods. There are thousands of publications in face recognition using holistic approaches. And generally this type of approaches can achieve better performance than feature based approaches [2], [3]. However, the performance of holistic matching methods will drop when there are variations due to expressions or poses. And local features extracted from local regions of a face image are more robust to these variations than the global features. This motivates us to re-study the feature based approaches, and compare them with the most popular holistic approaches: Fisherface (FLDA), the null space approach (NLDA) and Eigenfeature Regularization and Extraction (ERE). Recently, Scale Invariant Feature Transform (SIFT) has been successfully applied in various general object recognition tasks [4], [5]. This approach extracts blob-like shape local features from an image, and represents each blob structure at the appropriate scale with explicit mechanism of automatic scale selection [6]. Even though there have been a few studies using SIFT for face recognition [7], [8], they simply applied the core SIFT algorithm. It is known that SIFT was designed for general object recognition/detection. General objects are rigid, and there are sharp transitions between different sides. However, faces are non-rigid and smooth. There are less structures with high contrast or high edge responses. Hence, the original SIFT approach may not be optimal for face recognition applications. In this paper we analyze the SIFT approach and study its deficiencies when applied to face recognition applications. Based on this study, new approach (Volume-SIFT) to remove unreliable keypoints detected by SIFT algorithm is proposed. Furthermore, to keep keypoints detected at large scale and near face boundaries, we propose Partial-Descriptor-SIFT (PDSIFT) approach. Comparisons between feature based approaches and holistic approaches are also given. We evaluate all these approaches on two databases: ORL and AR. Experimental results show that our proposed approaches can achieve better performance than the original SIFT approach. Moreover, PDSIFT can achieve comparable performance as the most successful holistic approach Eigenfeature Regularization and Extraction (ERE) [9], and significantly outperform Fisherface (FLDA) [10] and the null space approach (NLDA) [11]. As described in [5], SIFT algorithm is composed of four stages of computation: 1. Scale-space extrema detection. To detect blob structures in an image, a scale space is constructed where the interest points, which are called keypoints in the SIFT framework, are detected. The scale space function is produced from the convolution of a variable-scale Gaussian, G(x, y, σ2), with an input image

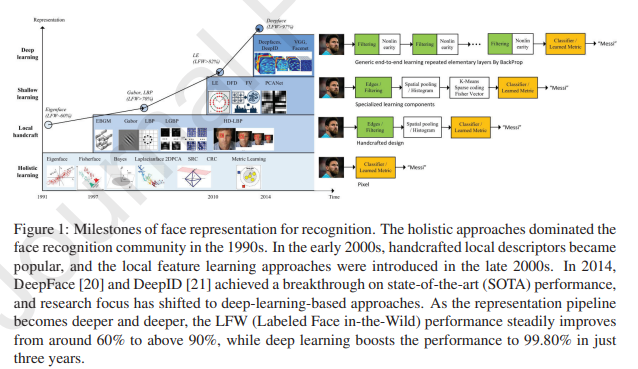
**IN “ROHIT SATLE, VISHNUPRASAD POOJARY, JOHN ABRAHAM, SHILPA WAKODE, "MISSING CHILD IDENTIFICATION USING FACE RECOGNITION SYSTEM", INTERNATIONAL JOURNAL OF ADVANCED ENGINEERING AND INNOVATIVE TECHNOLOGY (IJAEIT), VOLUME 3 ISSUE 1 JULY - AUGUST 2016”**

The human face plays an important role in our social interaction, conveying people’s identity. Face recognition is a task that humans perform routinely and effortlessly in their daily lives. Face recognition, as one of the primary biometric technologies, became more and more important owing to rapid advances in technologies such as digital cameras, the Internet and mobile devices, and increased demands on security. A facial recognition system is a computer application capable of identifying or verifying a person from a digital image or a video frame from a video source. Face Recognition System is a computer based digital technology and is an active area of research. This paper addresses the building of face recognition system by using Principal Component Analysis (PCA) method. The PCA has been extensively employed for face recognition algorithms. It not only reduces the dimensionality of the image, but also retains some of the variations in the image data. The system functions by projecting face image onto a feature space that spans the significant variations among known face images. The significant features are known as “Eigen faces”, because they are the eigenvectors (Principal Component) of the set of faces they do not necessarily correspond to the features such as eyes, ears, and noses. The projection operation characterize an individual face by a weighted sum of the Eigen faces features and so to recognize a particular face it is necessary only to compare these weights to those individuals. Biometric-based techniques have emerged as the most promising option for recognizing individuals in recent years since, instead of authenticating people and granting them access to physical and virtual domains based on passwords, PINs, smart cards, plastic cards, tokens, keys and so forth, these methods examine an individual’s physiological and/or behavioral characteristics in order to determine and/or ascertain his identity. Passwords and PINs are hard to remember and can be stolen or guessed; cards, tokens, keys and the like can be misplaced, forgotten, purloined or duplicated; magnetic cards can become corrupted and unreadable. However, an individual’s biological traits cannot be misplaced, forgotten, stolen or forged. Biometric-based technologies include identification based on physiological characteristics (such as face, fingerprints, finger geometry, hand geometry, hand veins, palm, iris, retina, ear and voice) and behavioral traits (such as gait, signature and keystroke dynamics). Face recognition appears to offer several advantages over other biometric methods, a few of which are outlined here: Almost all these technologies require some voluntary action by the user, i.e., the user needs to place his hand on a hand-rest for fingerprinting or hand geometry detection and has to stand in a fixed position in front of a camera for iris or retina identification. However, face recognition can be done passively without any explicit action or participation on the part of the user since face images can be acquired from a distance by a camera. This is particularly beneficial for security and surveillance purposes. Furthermore, data acquisition in general is fraught with problems for other biometrics: techniques that rely on hands and fingers can be rendered useless if the epidermis tissue is damaged in some way (i.e., bruised or cracked). Iris and retina identification require expensive equipment and are much too sensitive to any body motion. Voice recognition is susceptible to background noises in public places and auditory fluctuations on a phone line or tape recording. Signatures can be modified or forged. However, facial images can be easily obtained with a couple of inexpensive fixed cameras. Good face recognition algorithms and appropriate preprocessing of the images can compensate for noise and slight variations in orientation, scale and illumination. Finally, technologies that require multiple individuals to use the same equipment to capture their biological characteristics potentially expose the user to the transmission of germs and impurities from other users. However, face recognition is totally nonintrusive and does not carry any such health risks.

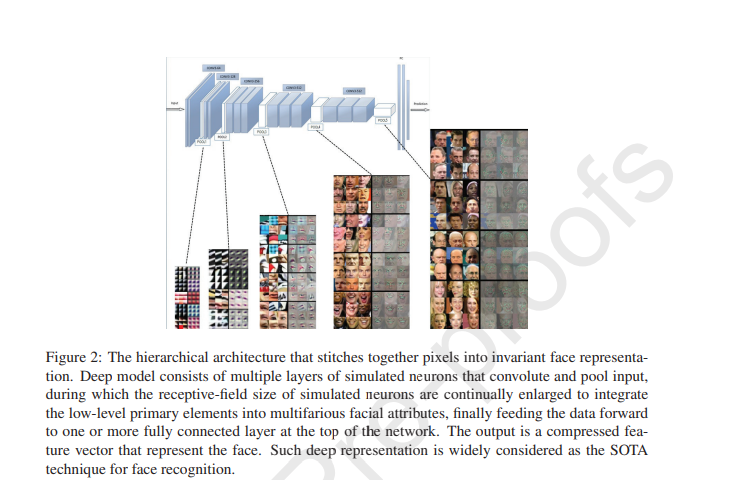
**IN “MISSING CHILD IDENTIFICATION USING FACE AUTHENTICATION USING KNN”** In India a countless number of children are reported missing every year. Among the missing child cases a large percentage of children remain untraced. This paper presents a novel use of deep learning methodology for identifying the reported missing child from the photos of multitude of children available, with the help of face recognition. The public can upload photographs of suspicious child into a common portal with landmarks and remarks. The photo will be automatically compared with the registered photos of the missing child from the repository. Classification of the input child image is performed and photo with best match will be selected from the database of missing children. For this, a deep learning model is trained to correctly identify the missing child from the missing child image database provided, using the facial image uploaded by the public. The Convolutional Neural Network (CNN), a highly effective deep learning technique for image based applications is adopted here for face recognition. Face descriptors are extracted from the images using a pre-trained CNN model VGG-Face deep architecture. Compared with normal deep learning applications, our algorithm uses convolution network only as a high level feature extractor and the child recognition is done by the trained KNN classifier. Choosing the best performing CNN model for face recognition, VGG-Face and proper training of it results in a deep learning model invariant to noise, illumination, contrast, occlusion, image pose and age of the child and it outperforms earlier methods in face recognition based missing child identification Children are the greatest asset of each nation. The future of any country depends upon the right upbringing of its children. India is the second populous country in the world and children represent a significant percentage of total population. But unfortunately a large number of children go missing every year in India due to various reasons including abduction or kidnapping, run-away children, trafficked children and lost children. A deeply disturbing fact about India’s missing children is that while on an average 174 children go missing every day, half of them remain untraced. Children who go missing may be exploited and abused for various purposes. As per the National Crime Records Bureau (NCRB) report which was cited by the Ministry of Home Affairs (MHA) in the Parliament (LS Q no. 3928, 20-032018), more than one lakh children (1,11,569 in actual numbers) were reported to have gone missing till 2016, and 55,625 of them remained untraced till the end of the year. Many NGOs claim that estimates of missing children are much higher than reported. The missing from one region may be found in another region or another state, for various reasons. So even if a child is found, it is difficult to identify him/her from the reported missing cases. A framework and methodology for developing an assistive tool for tracing missing child is described in this paper. An idea for maintaining a virtual space is proposed, such that the recent photographs of children given by parents at the time of reporting missing cases is saved in a repository. The public is given provision to voluntarily take photographs of children in suspected situations and uploaded in that portal. Automatic searching of this photo among the missing child case images will be provided in the application. This supports the police officials to locate the child anywhere in India. Mostly missing child cases are reported to the police. The child missing from one region may be found in another region or another state, for various reasons. So even if a child is found, it is difficult to identify him/her from the reported missing cases. A framework and methodology for developing an assistive tool for tracing missing child is described in this paper. An idea for maintaining a virtual space is proposed, such that the recent photographs of children given by parents at the time of reporting missing cases is saved in a repository. The public is given provision to voluntarily take photographs of children in suspected situations and uploaded in that portal. Automatic searching of this photo among the missing child case images will be provided in the application. This supports the police officials to locate the child anywhere in India Here we propose a methodology for missing child identification which combines facial feature extraction based on deep learning and matching based on KNN. The proposed system utilizes face recognition for missing child identification. This is to help authorities and parents in missing child investigation.

**IN “SIMONYAN, KAREN AND ANDREW ZISSERMAN, "VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION", INTERNATIONAL CONFERENCE ON LEARNING REPRESENTATIONS ( ICLR), APRIL 2015.”** In this work we investigate the effect of the convolutional network depth on its accuracy in the large-scale image recognition setting. Our main contribution is a thorough evaluation of networks of increasing depth using an architecture with very small ( 3 × 3) convolution filters, which shows that a significant improvement on the prior-art configurations can be achieved by pushing the depth to 16–19 weight layers. These findings were the basis of our ImageNet Challenge 2014 submission, where our team secured the first and the second places in the localisation and classification tracks respectively. We also show that our representations generalise well to other datasets, where they achieve state-of-the-art results. We have made our two best-performing ConvNet models publicly available to facilitate further research on the use of deep visual representations in computer vision. Convolutional networks (ConvNets) have recently enjoyed a great success in large-scale image and video recognition (Krizhevsky et al., 2012; Zeiler & Fergus, 2013; Sermanet et al., 2014; Simonyan & Zisserman, 2014) which has become possible due to the large public image repositories, such as ImageNet (Deng et al., 2009), and high-performance computing systems, such as GPUs or large-scale distributed clusters (Dean et al., 2012). In particular, an important role in the advance of deep visual recognition architectures has been played by the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) (Russakovsky et al., 2014), which has served as a testbed for a few generations of large-scale image classification systems, from high-dimensional shallow feature encodings (Perronnin et al., 2010) (the winner of ILSVRC-2011) to deep ConvNets (Krizhevsky et al., 2012) (the winner of ILSVRC-2012). With ConvNets becoming more of a commodity in the computer vision field, a number of attempts have been made to improve the original architecture of Krizhevsky et al. (2012) in a bid to achieve better accuracy. For instance, the best-performing submissions to the ILSVRC2013 (Zeiler & Fergus, 2013; Sermanet et al., 2014) utilised smaller receptive window size and smaller stride of the first convolutional layer. Another line of improvements dealt with training and testing the networks densely over the whole image and over multiple scales (Sermanet et al., 2014; Howard, 2014). In this paper, we address another important aspect of ConvNet architecture design – its depth. To this end, we fix other parameters of the architecture, and steadily increase the depth of the network by adding more convolutional layers, which is feasible due to the use of very small ( 3 × 3) convolution filters in all layers. As a result, we come up with significantly more accurate ConvNet architectures, which not only achieve the state-of-the-art accuracy on ILSVRC classification and localisation tasks, but are also applicable to other image recognition datasets, where they achieve excellent performance even when used as a part of a relatively simple pipelines (e.g. deep features classified by a linear SVM without fine-tuning). We have released our two best-performing models 1 to facilitate further research

**IN “O. M. PARKHI, A. VEDALDI, AND A. ZISSERMAN, "DEEP FACE RECOGNITION," IN BRITISH MACHINE VISION CONFERENCE, VOL. 1, NO. 3, PP. 1-12, 2015”** Deep learning applies multiple processing layers to learn representations of data with multiple levels of feature extraction. This emerging technique has reshaped the research landscape of face recognition (FR) since 2014, launched by the breakthroughs of DeepFace and DeepID. Since then, deep learning technique, characterized by the hierarchical architecture to stitch together pixels into invariant face representation, has dramatically improved the state-of-the-art performance and fostered successful real-world applications. In this survey, we provide a comprehensive review of the recent developments on deep FR, covering broad topics on algorithm designs, databases, protocols, and application scenes. First, we summarize different network architectures and loss functions proposed in the rapid evolution of the deep FR methods. Second, the related face processing methods are categorized into two classes: “one-to-many augmentation” and “many-to-one normalization”. Then, we summarize and compare the commonly used databases for both model training and evaluation. Third, we review miscellaneous scenes in deep FR, such as cross-factor, heterogenous, multiple-media and industrial scenes. Finally, the technical challenges and several promising directions are highlighted Face recognition (FR) has been the prominent biometric technique for identity authentication and has been widely used in many areas, such as military, fi- nance, public security and daily life. FR has been a long-standing research topic in the CVPR community. In the early 1990s, the study of FR became popular following the introduction of the historical Eigenface approach [1]. The milestones of feature-based FR over the past years are presented in Fig. 1, in which the times of four major technical streams are highlighted. The holistic approaches derive the low-dimensional representation through certain distribution assumptions, such as linear subspace [2][3][4], manifold [5][6][7], and sparse representation [8][9][10][11]. This idea dominated the FR community in the 1990s and 2000s. However, a well-known problem is that these theoretically plausible holistic methods fail to address the uncontrolled facial changes that deviate from their prior assumptions. In the early 2000s, this problem gave rise to local-feature-based FR. Gabor [12] and LBP [13], as well as their multilevel and high-dimensional extensions [14][15][16], achieved robust performance through some invariant properties of local filtering. Unfortunately, handcrafted features suffered from a lack of distinctiveness and compactness. In the early 2010s, learning-based local descriptors were introduced to the FR community [17][18][19], in which local filters are learned for better distinctiveness and the encoding codebook is learned for better compactness. However, these shallow representations still have an inevitable limitation on robustness against the complex nonlinear facial appearance variations.



In general, traditional methods attempted to recognize human face by one or two layer representations, such as filtering responses, histogram of the feature codes, or distribution of the dictionary atoms. The research community studied intensively to separately improve the preprocessing, local descriptors, and feature transformation, but these approaches improved FR accuracy slowly. What’s worse, most methods aimed to address one aspect of unconstrained facial changes only, such as lighting, pose, expression, or disguise. There was no any integrated technique to address these unconstrained challenges integrally. As a result, with continuous efforts of more than a decade, “shallow” methods only improved the accuracy of the LFW benchmark to about 95% [15], which indicates that “shallow” methods are insufficient to extract stable identity feature invariant to real-world changes. Due to the insufficiency of this technical, facial recognition systems were often reported with unstable performance or failures with countless false alarms in real-world applications. But all that changed in 2012 when AlexNet won the ImageNet competition by a large margin using a technique called deep learning [22]. Deep learning methods, such as convolutional neural networks, use a cascade of multiple layers of processing units for feature extraction and transformation. They learn multiple levels of representations that correspond to different levels of abstraction. The levels form a hierarchy of concepts, showing strong invariance to the face pose, lighting, and expression changes, as shown in Fig. 2. It can be seen from the figure that the first layer of the deep neural network is somewhat similar to the Gabor feature found by human scientists with years of experience. The second layer learns more complex texture features. The features of the third layer are more complex, and some simple structures have begun to appear, such as highbridged nose and big eyes. In the fourth, the network output is enough to explain a certain facial attribute, which can make a special response to some clear abstract concepts such as smile, roar, and even blue eye. In conclusion, in deep convolutional neural networks (CNN), the lower layers automatically learn the features similar to Gabor and SIFT designed for years or even decades (such as initial layers in Fig. 2), and the higher layers further learn higher level abstraction. Finally, the combination of these higher level abstraction represents facial identity with unprecedented stability. In 2014, DeepFace [20] achieved the SOTA accuracy on the famous LFW benchmark [23], approaching human performance on the unconstrained condition for the first time (DeepFace: 97.35% vs. Human: 97.53%), by training a 9-layer model on 4 million facial images. Inspired by this work, research focus has shifted to deep-learning-based approaches, and the accuracy was dramatically boosted to above 99.80% in just three years. Deep learning technique has reshaped the



research landscape of FR in almost all aspects such as algorithm designs, training/test datasets, application scenarios and even the evaluation protocols. Therefore, it is of great significance to review the breakthrough and rapid development process in recent years. There have been several surveys on FR [24, 25, 26, 27, 28] and its subdomains, and they mostly summarized and compared a diverse set of techniques related to a specific FR scene, such as illumination-invariant FR [29], 3D FR [28], pose-invariant FR [30][31]. Unfortunately, due to their earlier publication dates, none of them covered the deep learning methodology that is most successful nowadays. This survey focuses only on recognition problem, and one can refer to Ranjan et al. [32] for a brief review of a full deep FR pipeline with detection and alignment, or refer to Jin et al. [33] for a survey of face alignment. Specifically, the major contributions of this survey are as follows: • A systematic review on the evolution of the network architectures and loss functions for deep FR is provided. Various loss functions are categorized into Euclidean-distance-based loss, angular/cosine-margin-based loss and softmax loss and its variations. Both the mainstream network architectures, such as Deepface [20], DeepID series [34, 35, 21, 36], VGGFace [37], FaceNet [38], and VGGFace2 [39], and other architectures designed for FR are covered. • We categorize the new face processing methods based on deep learning, such as those used to handle recognition difficulty on pose changes, into two classes: “one-to-many augmentation” and “many-to-one normalization”, and discuss how emerging generative adversarial network (GAN) [40] facilitates deep FR. • We present a comparison and analysis on public available databases that are of vital importance for both model training and testing. Major FR benchmarks, such as LFW [23], IJB-A/B/C [41, 42, 43], Megaface [44], and MSCeleb-1M [45], are reviewed and compared, in term of the four aspects: training methodology, evaluation tasks and metrics, and recognition scenes, which provides an useful reference for training and testing deep FR. • Besides the general purpose tasks defined by the major databases, we summarize a dozen scenario-specific databases and solutions that are still challenging for deep learning, such as anti-attack, cross-pose FR, and cross-age FR. By reviewing specially designed methods for these unsolved problems, we attempt to reveal the important issues for future research on deep FR, such as adversarial samples, algorithm/data biases, and model interpretability.

**IN “A. VEDALDI, AND K. LENC, "MATCONVNET: CONVOLUTIONAL NEURAL NETWORKS FOR MATLAB", ACM INTERNATIONAL CONFERENCE ON MULTIMEDIA, BRISBANE, OCTOBER 2015.”** MatConvNet is an implementation of Convolutional Neural Networks (CNNs) for MATLAB. The toolbox is designed with an emphasis on simplicity and flexibility. It exposes the building blocks of CNNs as easy-to-use MATLAB functions, providing routines for computing linear convolutions with filter banks, feature pooling, and many more. In this manner, MatConvNet allows fast prototyping of new CNN architectures; at the same time, it supports efficient computation on CPU and GPU allowing to train complex models on large datasets such as ImageNet ILSVRC. This document provides an overview of CNNs and how they are implemented in MatConvNet and gives the technical details of each computational block in the toolbox. MatConvNet is a MATLAB toolbox implementing Convolutional Neural Networks (CNN) for computer vision applications. Since the breakthrough work of [8], CNNs have had a major impact in computer vision, and image understanding in particular, essentially replacing traditional image representations such as the ones implemented in our own VLFeat [13] open source library. While most CNNs are obtained by composing simple linear and non-linear filtering operations such as convolution and rectification, their implementation is far from trivial. The reason is that CNNs need to be learned from vast amounts of data, often millions of images, requiring very efficient implementations. As most CNN libraries, MatConvNet achieves this by using a variety of optimizations and, chiefly, by supporting computations on GPUs. Numerous other machine learning, deep learning, and CNN open source libraries exist. To cite some of the most popular ones: CudaConvNet,1 Torch,2 Theano,3 and Caffe4 . Many of these libraries are well supported, with dozens of active contributors and large user bases. Therefore, why creating yet another library? The key motivation for developing MatConvNet was to provide an environment particularly friendly and efficient for researchers to use in their investigations.5 MatConvNet achieves this by its deep integration in the MATLAB environment, which is one of the most popular development environments in computer vision research as well as in many other areas. In particular, MatConvNet exposes as simple MATLAB commands CNN building blocks such as convolution, normalisation and pooling (chapter 4); these can then be combined and extended with ease to create CNN architectures. While many of such blocks use optimised CPU and GPU implementations written in C++ and CUDA (section section 1.4), MATLAB native support for GPU computation means that it is often possible to write new blocks in MATLAB directly while maintaining computational efficiency. Compared to writing new CNN components using lower level languages, this is an important simplification that can significantly accelerate testing new ideas. Using MATLAB also provides a bridge towards other areas; for instance, MatConvNet was recently used by the University of Arizona in planetary science, as summarised in this NVIDIA blogpost.6 MatConvNet can learn large CNN models such AlexNet [8] and the very deep networks of [11] from millions of images. Pre-trained versions of several of these powerful models can be downloaded from the MatConvNet home page7 . While powerful, MatConvNet remains simple to use and install. The implementation is fully self-contained, requiring only MATLAB and a compatible C++ compiler (using the GPU code requires the freely-available CUDA DevKit and a suitable NVIDIA GPU). As demonstrated in fig. 1.1 and section 1.1, it is possible to download, compile, and install MatConvNet using three MATLAB commands. Several fully-functional examples demonstrating how small and large networks can be learned are included. Importantly, several standard pre-trained network can be immediately downloaded and used in applications. A manual with a complete technical description of the toolbox is maintained along with the toolbox.8 These features make MatConvNet useful in an educational context too MatConvNet has a simple design philosophy. Rather than wrapping CNNs around complex layers of software, it exposes simple functions to compute CNN building blocks, such as linear convolution and ReLU operators, directly as MATLAB commands. These building blocks are easy to combine into complete CNNs and can be used to implement sophisticated learning algorithms. While several real-world examples of small and large CNN architectures and training routines are provided, it is always possible to go back to the basics and build your own, using the efficiency of MATLAB in prototyping. Often no C coding is required at all to try new architectures. As such, MatConvNet is an ideal playground for research in computer vision and CNNs. MatConvNet contains the following elements:  CNN computational blocks. A set of optimized routines computing fundamental building blocks of a CNN. For example, a convolution block is implemented by y=vl\_nnconv(x,f,b) where x is an image, f a filter bank, and b a vector of biases (section 4.1). The derivatives are computed as [dzdx,dzdf,dzdb] = vl\_nnconv(x,f,b,dzdy) where dzdy is the derivative of the CNN output w.r.t y (section 4.1). chapter 4 describes all the blocks in detail.  CNN wrappers. MatConvNet provides a simple wrapper, suitably invoked by vl\_simplenn, that implements a CNN with a linear topology (a chain of blocks). It also provides a much more flexible wrapper supporting networks with arbitrary topologies, encapsulated in the dagnn.DagNN MATLAB class.  Example applications. MatConvNet provides several examples of learning CNNs with stochastic gradient descent and CPU or GPU, on MNIST, CIFAR10, and ImageNet data.  Pre-trained models. MatConvNet provides several state-of-the-art pre-trained CNN models that can be used off-the-shelf, either to classify images or to produce image encodings in the spirit of Caffe or DeCAF. There are three main sources of information about MatConvNet. First, the website contains descriptions of all the functions and several examples and tutorials.11 Second, there is a PDF manual containing a great deal of technical details about the toolbox, including detailed mathematical descriptions of the building blocks. Third, MatConvNet ships with several examples (section 1.1). Most examples are fully self-contained. For example, in order to run the MNIST example, it suffices to point MATLAB to the MatConvNet root directory and type addpath ←- examples followed by cnn\_mnist. Due to the problem size, the ImageNet ILSVRC example requires some more preparation, including downloading and preprocessing the images (using the bundled script utils/preprocess−imagenet.sh). Several advanced examples are included as well. For example, fig. 1.2 illustrates the top-1 and top-5 validation errors as a model similar to AlexNet [8] is trained using either standard dropout regularisation or the recent batch normalisation technique of [4]. The latter is shown to converge in about one third of the epochs (passes through the training data) required by the former. The MatConvNet website contains also numerous pre-trained models, i.e. large CNNs trained on ImageNet ILSVRC that can be downloaded and used as a starting point for many other problems [1]. These include: AlexNet [8], VGG-S, VGG-M, VGG-S [1], and VGG-VD16, and VGG-VD-19 [12]. The example code of fig. 1.1 shows how one such model can be used in a few lines of MATLAB code Efficiency is very important for working with CNNs. MatConvNet supports using NVIDIA GPUs as it includes CUDA implementations of all algorithms (or relies on MATLAB CUDA support). To use the GPU (provided that suitable hardware is available and the toolbox has been compiled with GPU support), one simply converts the arguments to gpuArrays in MATLAB, as in y = vl\_nnconv(gpuArray(x), gpuArray(w), []). In this manner, switching between CPU and GPU is fully transparent. Note that MatConvNet can also make use of the NVIDIA CuDNN library with significant speed and space benefits. Next we evaluate the performance of MatConvNet when training large architectures on the ImageNet ILSVRC 2012 challenge data [2]. The test machine is a Dell server with two Intel Xeon CPU E5-2667 v2 clocked at 3.30 GHz (each CPU has eight cores), 256 GB of RAM, and four NVIDIA Titan Black GPUs (only one of which is used unless otherwise noted). Experiments use MatConvNet beta12, CuDNN v2, and MATLAB R2015a. The data is preprocessed to avoid rescaling images on the fly in MATLAB and stored in a RAM disk for faster access. The code uses the vl\_imreadjpeg command to read large batches of JPEG images from disk in a number of separate threads. The driver examples/cnn\_imagenet.m is used in all experiments. We train the models discussed in section 1.3 on ImageNet ILSVRC. table 1.1 reports the training speed as number of images per second processed by stochastic gradient descent. AlexNet trains at about 264 images/s with CuDNN, which is about 40% faster than the vanilla GPU implementation (using CuBLAS) and more than 10 times faster than using the CPUs. Furthermore, we note that, despite MATLAB overhead, the implementation speed is comparable to Caffe (they report 253 images/s with CuDNN and a Titan – a slightly slower GPU than the Titan Black used here). Note also that, as the model grows in size, the size of a SGD batch must be decreased (to fit in the GPU memory), increasing the overhead impact somewhat. table 1.2 reports the speed on VGG-VD-16, a very large model, using multiple GPUs. In this case, the batch size is set to 264 images. These are further divided in sub-batches of 22 images each to fit in the GPU memory; the latter are then distributed among one to four GPUs on the same machine. While there is a substantial communication overhead, training speed increases from 20 images/s to 45. Addressing this overhead is one of the medium term goals of the library.

**PROPOSED SYSTEM**

This paper presents a novel use of deep learning methodology for identifying the reported missing child from the photos of multitude of children available, with the help of face recognition. The public can upload photographs of suspicious child into a common portal with landmarks and remarks. The photo will be automatically compared with the registered photos of the missing child from the repository. Classification of the input child image is performed and photo with best match will be selected from the database of missing children. For this, a deep learning model is trained to correctly identify the missing child from the missing child image database provided, using the facial image uploaded by the public.

**ADVANTAGES:**

A deep learning architecture considering all these constrain is designed here.

The proposed system is comparatively an easy, inexpensive and reliable method compared to other biometrics like finger print and iris recognition systems.

**IMPLEMENTATION**

WORK FLOW OF FACE RECOGNITION Here we propose a methodology for missing child identification which combines facial feature extraction based on deep learning and matching based on support vector machine. The proposed system utilizes face recognition for missing child identification. This is to help authorities and parents in missing child investigation. The architecture of the proposed frame work is given below

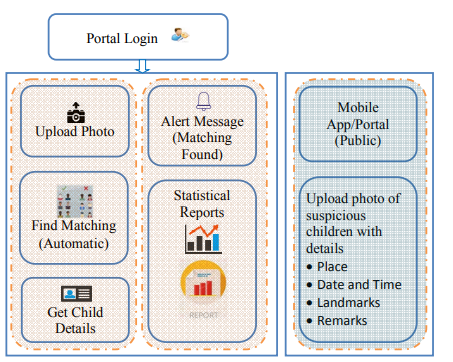


Fig: Architecture of proposed child identification system

It consists of a national portal for storing details of missing child along with the photo. Whenever a child missing is reported, along with the FIR, the concerned officer uploads the photo of the missing child into the portal. Public can search for any matching child in the database for the images with them. The system will prompt the most matching cases. Once the matching is found, the officer can get the details of the child. The system also generates various statistical reports. The public can upload photo of any suspicious child at any time into the portal with details like place, time, landmarks and remarks. The photo uploaded by the public will be automatically compared with photos of the registered missing children and if a matching photo with sufficient score is found, then an alert message will be sent to the concerned officer. The message will also be visible in the message box of the concerned officer login screen. The portal for the public can also be maintained as a mobile app, where he or she can upload photo of suspicious children with details. In the mobile app, location of the person updating the photo will also be automatically recorded. Whenever public uploads photo of a suspected child, the system generates template vector of the facial features from the uploaded photo. If a matching is found in the repository, the system displays the most matched photo and pushes a message to the concerned Officer portal or SMSs the alert message of matching child. Similarly the Officer can check for any matching with the database at any time using the proposed system. In the following sections the paper details the work flow for child matching methodology. The flow chart of the automatic child face identification methodology is as shown in Fig 2

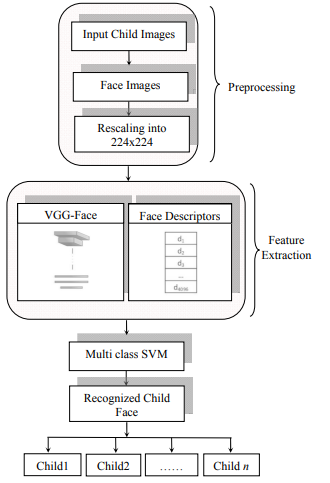


Fig: Software Flow of face recognition system

Images of reported missing children are saved in a repository and the face area is selected for cropping to obtain input face images. Learned features from a Convolutional Neural Network (CNN), a specific type of deep learning algorithm, are used for training a multi class SVM classifier. This machine learning approach is used to correctly label the child using the name indicated in the database provided by the concerned authority

**ALGORITHM**

# CONVOLUTIONAL NEURAL NETWORK

**Convolutional Neural Network** is one of the main categories to do image classification and image recognition in neural networks. Scene labeling, objects detections, and face recognition, etc., are some of the areas where convolutional neural networks are widely used.

CNN takes an image as input, which is classified and process under a certain category such as dog, cat, lion, tiger, etc. The computer sees an image as an array of pixels and depends on the resolution of the image. Based on image resolution, it will see as **h \* w \* d**, where h= height w= width and d= dimension. For example, An RGB image is **6 \* 6 \* 3** array of the matrix, and the grayscale image is **4 \* 4 \* 1** array of the matrix.

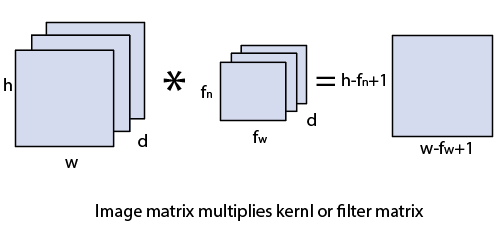
In CNN, each input image will pass through a sequence of convolution layers along with pooling, fully connected layers, filters (Also known as kernels). After that, we will apply the Soft-max function to classify an object with probabilistic values 0 and 1.



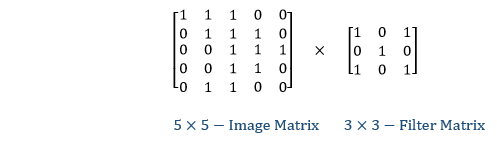
## Convolution Layer

Convolution layer is the first layer to extract features from an input image. By learning image features using a small square of input data, the convolutional layer preserves the relationship between pixels. It is a mathematical operation which takes two inputs such as image matrix and a kernel or filter.

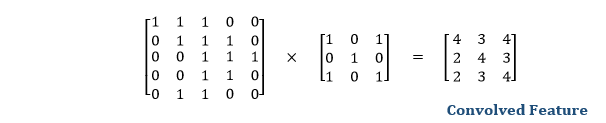
* The dimension of the image matrix is **h×w×d**.
* The dimension of the filter is **fh×fw×d**.
* The dimension of the output is **(h-fh+1)×(w-fw+1)×1**.



Let's start with consideration a 5\*5 image whose pixel values are 0, 1, and filter matrix 3\*3 as:



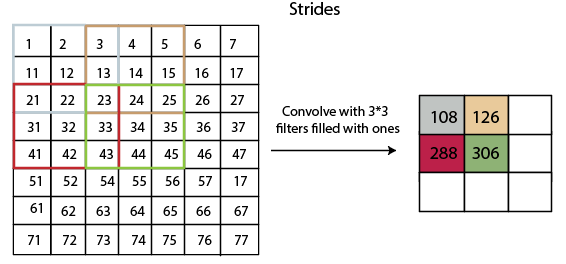
The convolution of 5\*5 image matrix multiplies with 3\*3 filter matrix is called "**Features Map**" and show as an output.



Convolution of an image with different filters can perform an operation such as blur, sharpen, and edge detection by applying filters.

## Strides

Stride is the number of pixels which are shift over the input matrix. When the stride is equaled to 1, then we move the filters to 1 pixel at a time and similarly, if the stride is equaled to 2, then we move the filters to 2 pixels at a time. The following figure shows that the convolution would work with a stride of 2.



## Padding

Padding plays a crucial role in building the convolutional neural network. If the image will get shrink and if we will take a neural network with 100's of layers on it, it will give us a small image after filtered in the end.

If we take a three by three filter on top of a grayscale image and do the convolving then what will happen?



It is clear from the above picture that the pixel in the corner will only get covers one time, but the middle pixel will get covered more than once. It means that we have more information on that middle pixel, so there are two downsides:

* Shrinking outputs
* Losing information on the corner of the image.

To overcome this, we have introduced padding to an image. **"Padding is an additional layer which can add to the border of an image."**

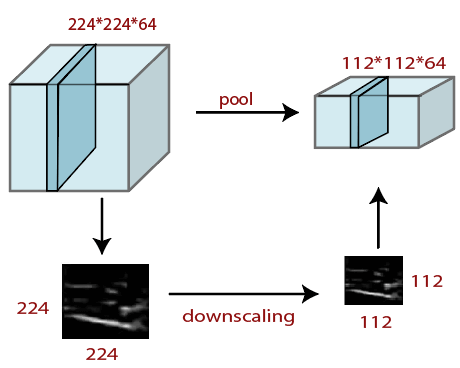
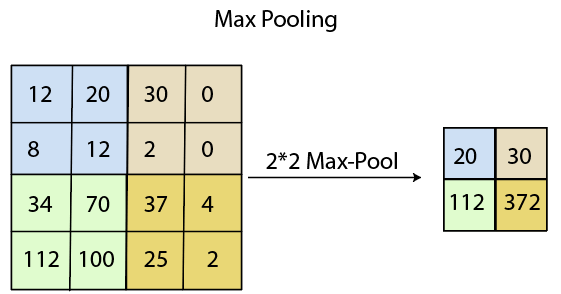
## Pooling Layer

Pooling layer plays an important role in pre-processing of an image. Pooling layer reduces the number of parameters when the images are too large. Pooling is "**downscaling**" of the image obtained from the previous layers. It can be compared to shrinking an image to reduce its pixel density. Spatial pooling is also called downsampling or subsampling, which reduces the dimensionality of each map but retains the important information. There are the following types of spatial pooling:

### Max Pooling

Max pooling is a **sample-based discretization process**. Its main objective is to downscale an input representation, reducing its dimensionality and allowing for the assumption to be made about features contained in the sub-region binned.

Max pooling is done by applying a max filter to non-overlapping sub-regions of the initial representation.



### Average Pooling

Down-scaling will perform through average pooling by dividing the input into rectangular pooling regions and computing the average values of each region.

**Syntax**

layer = averagePooling2dLayer(poolSize)

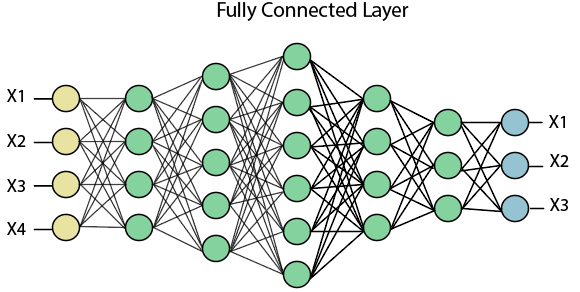
layer = averagePooling2dLayer(poolSize,Name,Value)

### Sum Pooling

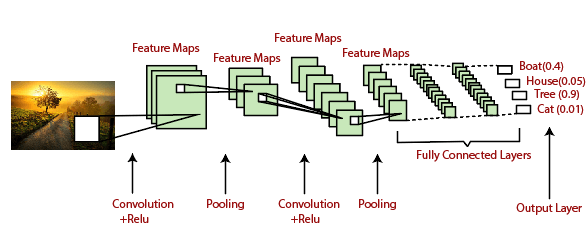
The sub-region for **sum pooling** or **mean pooling** are set exactly the same as for **max-pooling** but instead of using the max function we use sum or mean.

## Fully Connected Layer

The fully connected layer is a layer in which the input from the other layers will be flattened into a vector and sent. It will transform the output into the desired number of classes by the network.



In the above diagram, the feature map matrix will be converted into the vector such as **x1, x2, x3... xn** with the help of fully connected layers. We will combine features to create a model and apply the activation function such as **softmax** or **sigmoid** to classify the outputs as a car, dog, truck, etc.



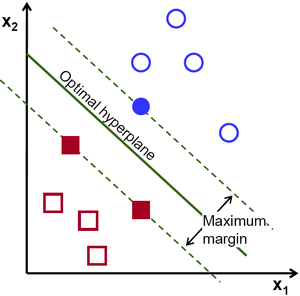
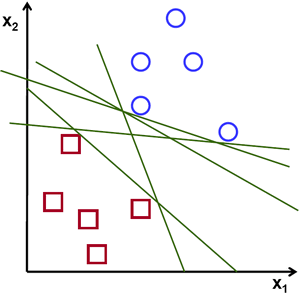
**VGG-Face CNN descriptor** A very deep CNN called VGG-Face network [8] is used for face recognition and its architecture is given in full detail in Fig 3. The CNN architecture comprises 11 blocks, each containing a linear operator followed by one or more nonlinearities such as ReLU and max pooling. The first eight such blocks are said to be convolutional as the linear operator is a bank of linear filters (linear convolution). It uses filters of size 3x3 with stride and pad of 1, throughout the network. All the convolution layers are followed by a rectification layer (ReLU). Max pooling layers used only 2x2 size with stride 2. The last three blocks are fully connected layers, they are the same as a convolutional layer, but the size of the filters matches the size of input data, such that each filter provides representative data from the entire image. Output of the first two FC layers are 4096 dimensional and the last FC layer has 2622 dimensions followed by L-dimensional metric embedding. Optimization is done by stochastic gradient descent using mini-batches of 64 samples and momentum coefficient of 0.9



Preprocessing input raw image in the context of face recognition involves acquiring the face region and standardizing images in a format compatible with the CNN architecture employed. Each CNN has a different input size requirement. The photographs of missing child acquired by a digital camera or mobile phone are taken and categorized into separate cases for creating the database of face recognition system. The face region in each image is identified and cropped for getting the input face images. The cropped face images are resized to 224x224 because VGGface network can process only RGB images in this particular size. The input to the deep network is fixed sized image with mean face image, computed from all the training set images, subtracted.

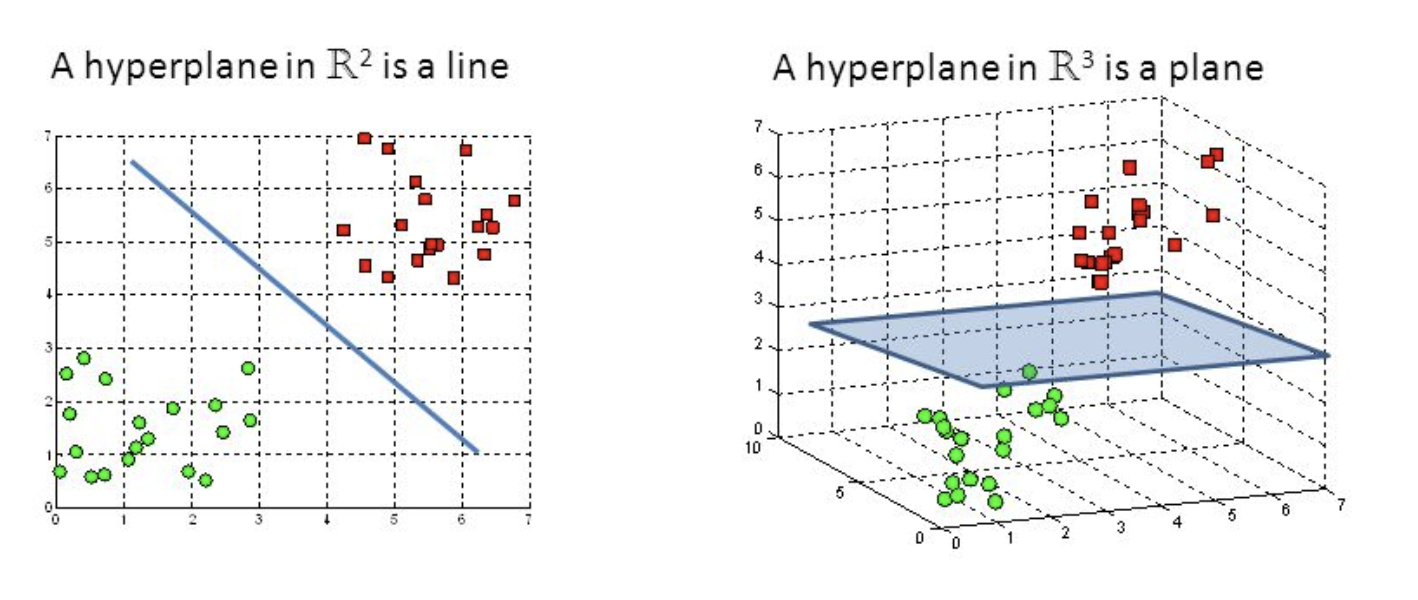
**SUPPORT VECTOR MACHINE?**

The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.

Possible hyperplanes

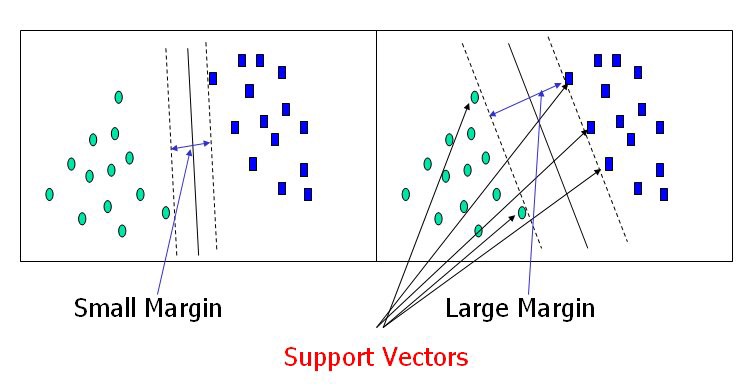
To separate the two classes of data points, there are many possible hyperplanes that could be chosen. Our objective is to find a plane that has the maximum margin, i.e the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.

Hyperplanes and Support Vectors



Hyperplanes in 2D and 3D feature space

Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes. Also, the dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3.



Support Vectors

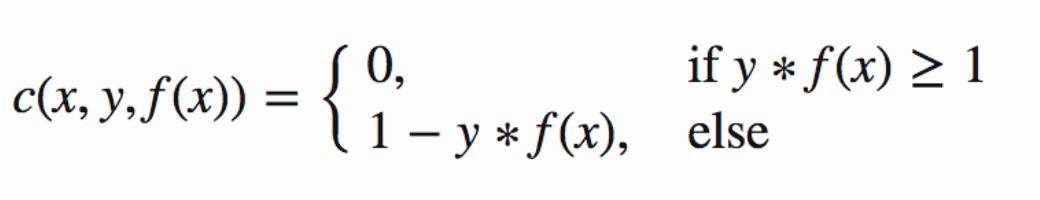
Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM.

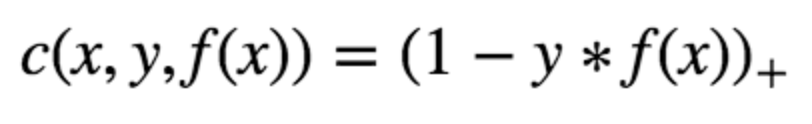
Large Margin Intuition

In logistic regression, we take the output of the linear function and squash the value within the range of [0,1] using the sigmoid function. If the squashed value is greater than a threshold value(0.5) we assign it a label 1, else we assign it a label 0. In SVM, we take the output of the linear function and if that output is greater than 1, we identify it with one class and if the output is -1, we identify is with another class. Since the threshold values are changed to 1 and -1 in SVM, we obtain this reinforcement range of values([-1,1]) which acts as margin.

Cost Function and Gradient Updates

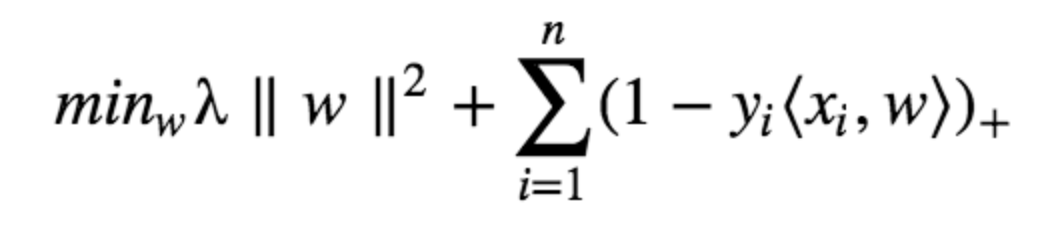
In the SVM algorithm, we are looking to maximize the margin between the data points and the hyperplane. The loss function that helps maximize the margin is hinge loss.





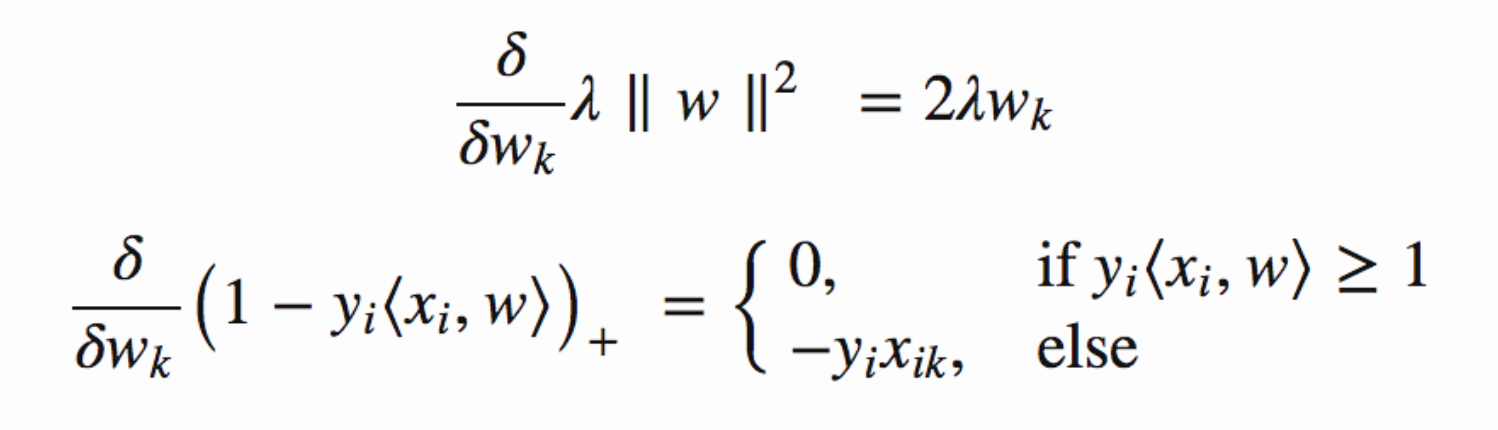
Hinge loss function (function on left can be represented as a function on the right)

The cost is 0 if the predicted value and the actual value are of the same sign. If they are not, we then calculate the loss value. We also add a regularization parameter the cost function. The objective of the regularization parameter is to balance the margin maximization and loss. After adding the regularization parameter, the cost functions looks as below.



Loss function for SVM

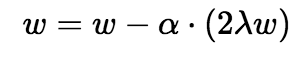
Now that we have the loss function, we take partial derivatives with respect to the weights to find the gradients. Using the gradients, we can update our weights.



Gradients

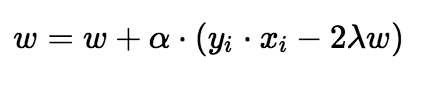
When there is no misclassification, i.e our model correctly predicts the class of our data point, we only have to update the gradient from the regularization parameter.





Gradient Update — No misclassification

When there is a misclassification, i.e our model make a mistake on the prediction of the class of our data point, we include the loss along with the regularization parameter to perform gradient update.



Gradient Update — Misclassification

**EXTRACTION OF FACIAL FEATURES** VGG-Face is trained to recognize the 2622 identities and other classes can’t be identified using this. But the activation vectors extracted from VGG-Face architecture can be used as the feature representations to classify each child category. The last classification layer is removed and extracts the 4K dimensional features from the first fully connected layers. The resulting feature vector is normalized by dividing each component by the L2 norm of this 4096 dimensional vector. Thus the pre-trained CNN VGG-Face is made to perform as an automatic facial feature extractor for training the classifier.

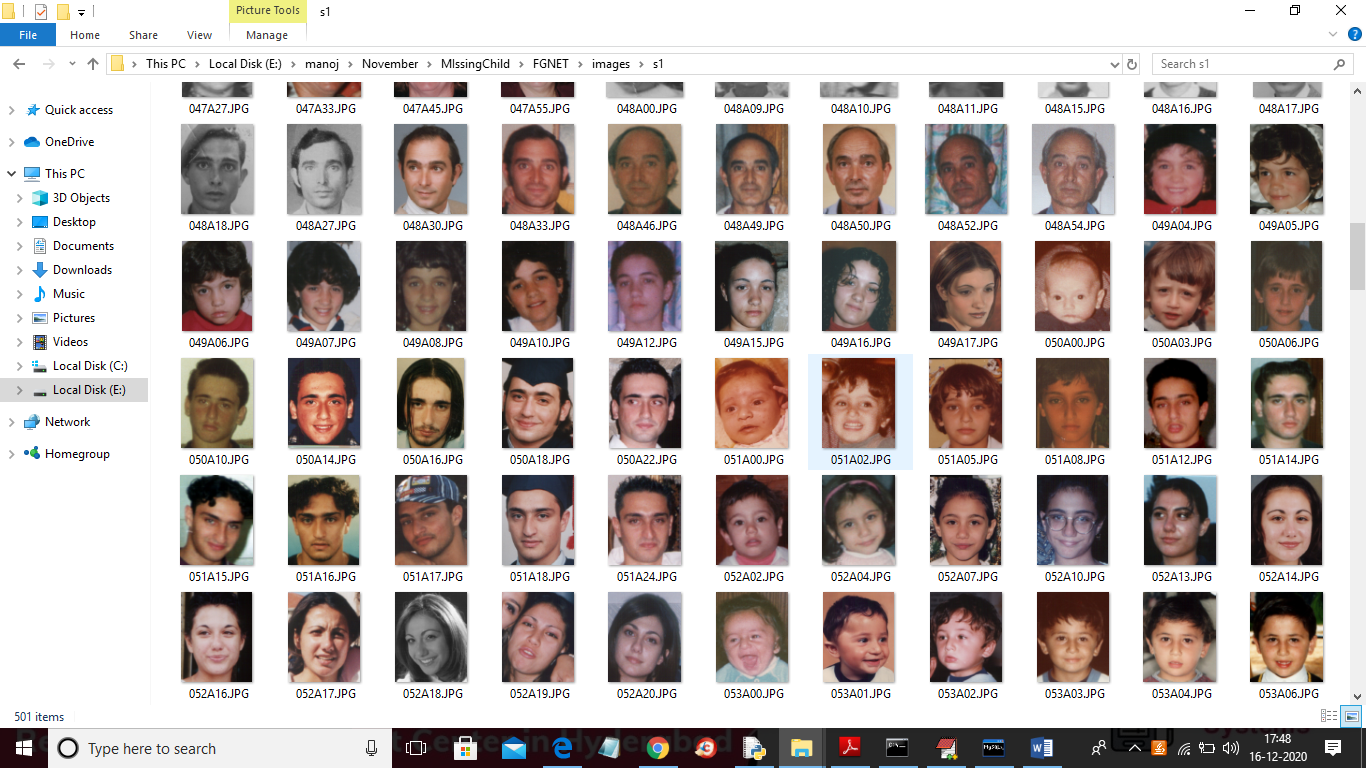
**MULTI CLASS SVM CLASSIFIER** Each face image corresponds to a child and child face recognition is considered as an image category classification problem. The task is to classify input image uploaded by the public into one of the given category based on the image representation. Basically CNN architecture consists of computational layers for feature extraction and a classifier layer at the final stage. The VGG-face CNN model employs the softmax activation function for labeled class prediction, suggesting the class each image belongs to. The softmax in the CNN layers is replaced with a multi class SVM trained with feature vector array from each image. One-versus-rest linear SVM classifier is used and is trained on the dataset. Extracted feature vector array is used to train this classifier

**SCREEN SHOTS**

In this paper author is describing concept to identify missing children by using Deep Learning and Multiclass SVM classifier and to implement this project author has used below modules

1. Using public dataset of missing children’s called FGNET is used to train deep learning CNN prediction model. After training model whenever public upload any suspected child image then this model will check in trained model to detect whether this child is in missing database or not. This detected result will store in database and whenever want official persons will login and see that detection result.
2. SVM Multiclass classifier use to extract face features from images based on age and other facial features and then this detected face will input to CNN model to predict whether this face child exists in image database or not.

First we used below dataset to train deep learning CNN model



To run project follow below steps

1) First create database in MYSQL by copying content from ‘DB.txt’ file and paste in MYQL

2) Install python, DJANGO and MYSQL software

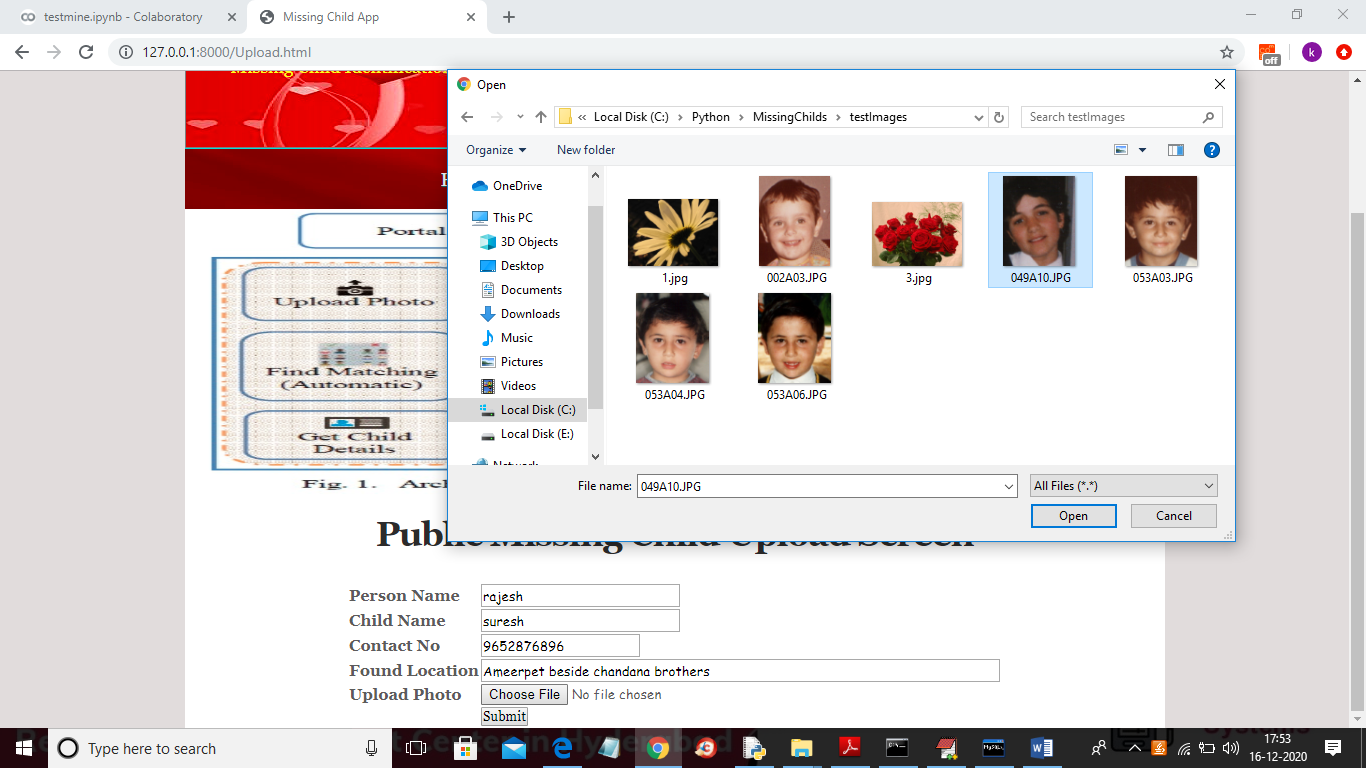
3) Create ‘Python’ folder in C directory and put ‘MissingChilds’ folder in it

4) start DJANGO server and run in browser to get first page

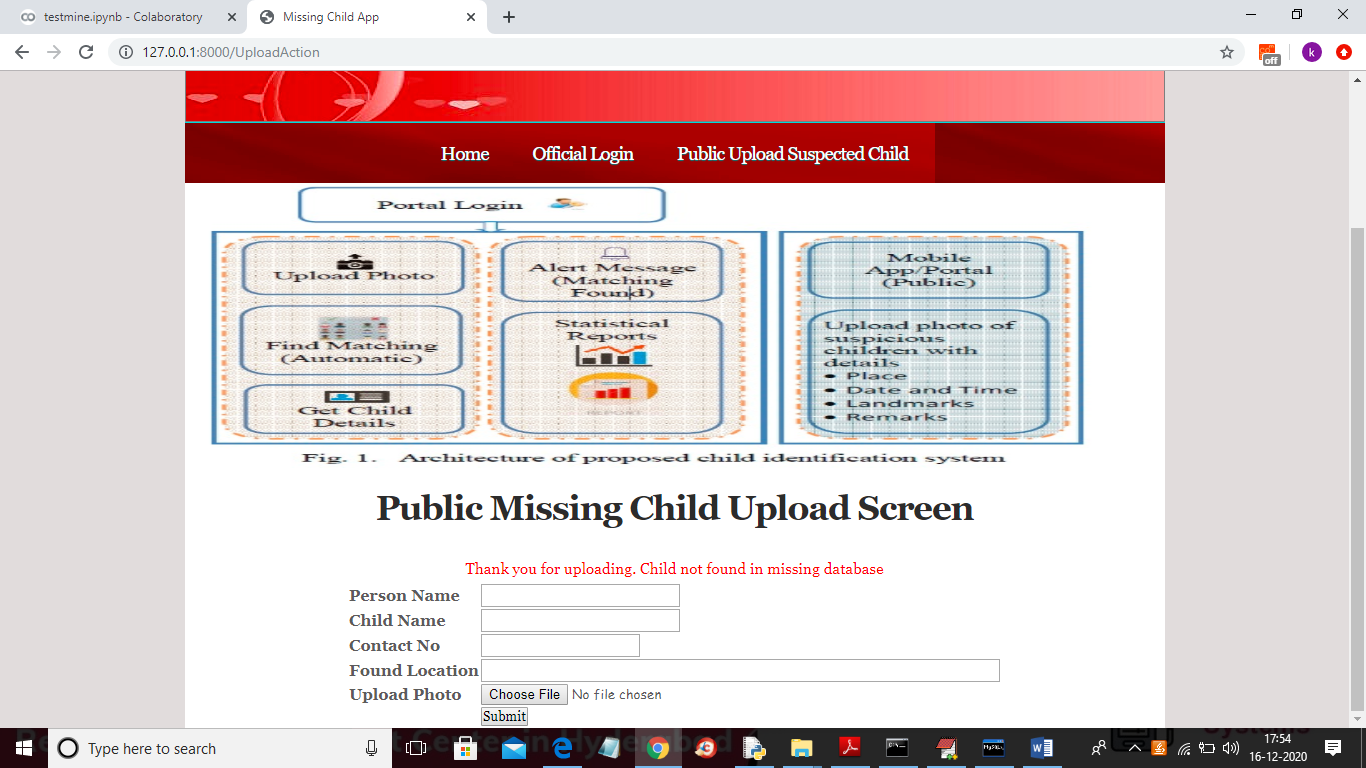
SCREEN SHOTS



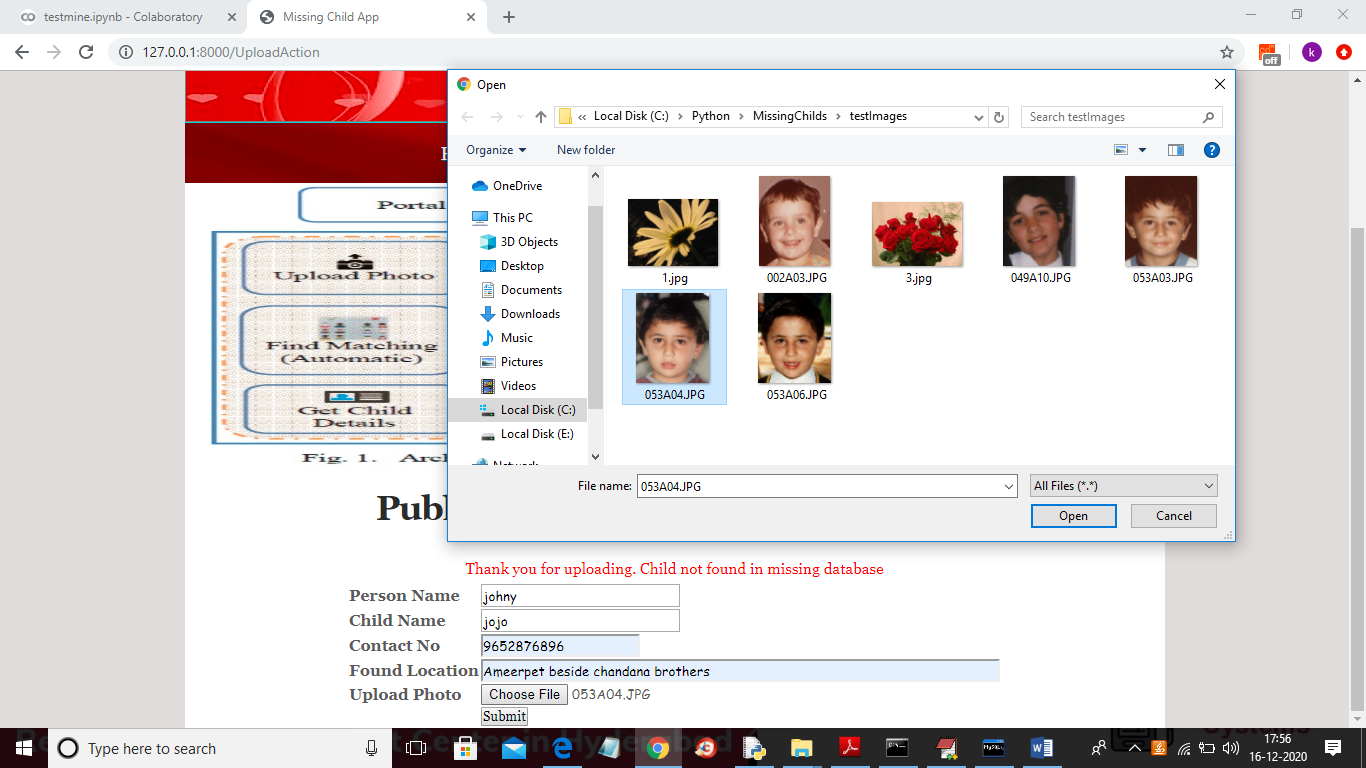
In above screen public can click on ‘Public Upload Suspected Child’ link to get below page and to add missing child details



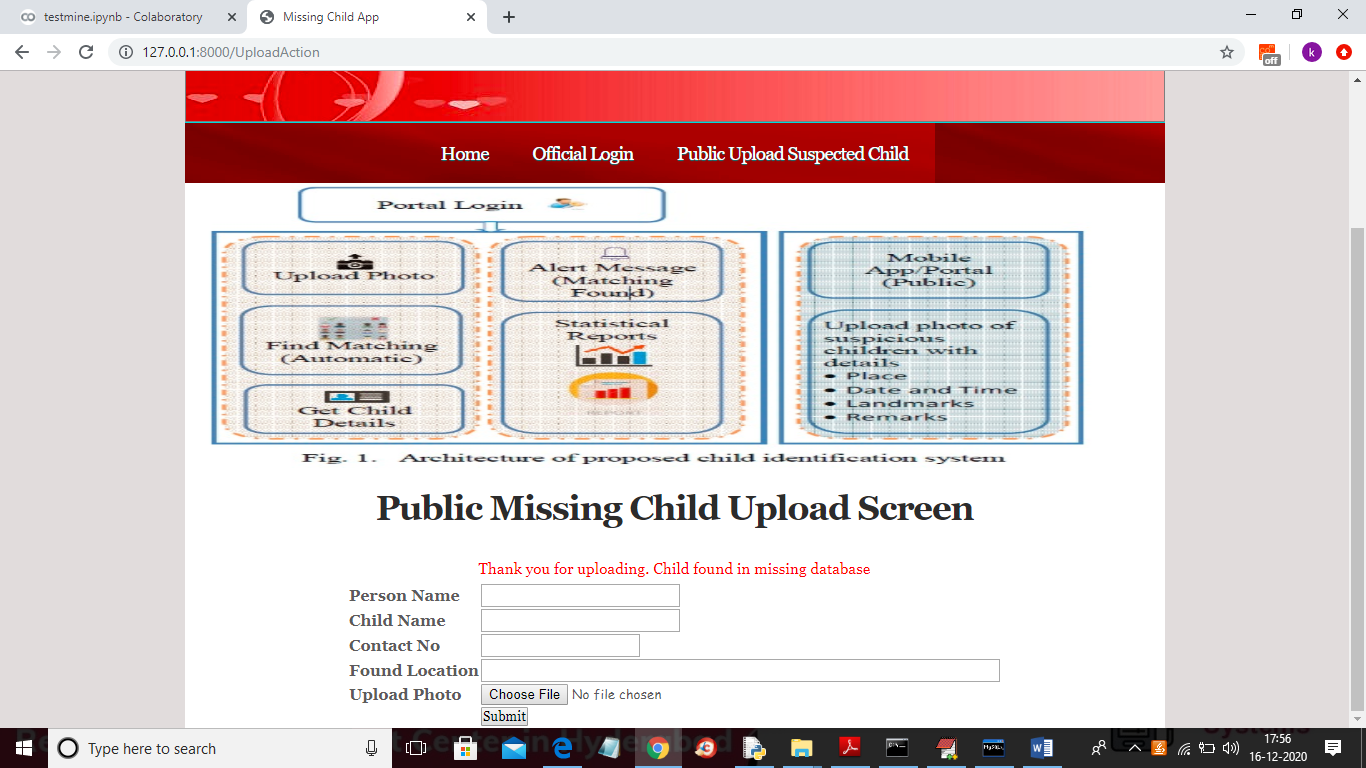
In above screen public will enter suspected child details and then upload photo and then click on ‘Submit’ button and to get below result



In above screen we can see child not found in missing DB and we can try with other image



And below is the result for new above child details



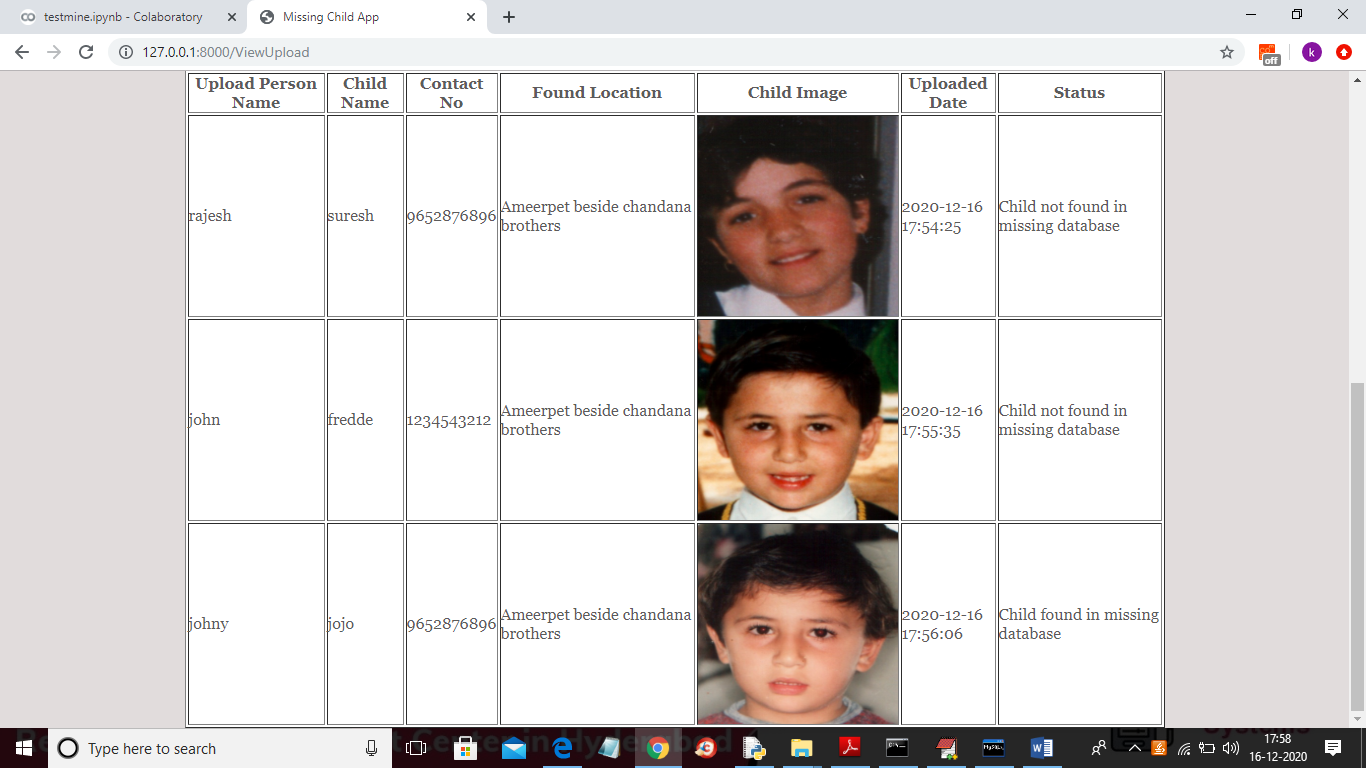
In above screen uploaded child found in database and now click on ‘Official Login’ link to get below login screen



In above screen admin can login by entering username and password as ‘admin’ and ‘admin’ and after clicking on ‘Login’ button will get below screen



In above screen official can click on ‘View Public Upload Missing Childs Status’ link to view all uploads and its result done by public



In above screen officials can see all details and then take action to find that child

**CONCLUSION** A missing child identification system is proposed, which combines the powerful CNN based deep learning approach for feature extraction and support vector machine classifier for classification of different child categories. This system is evaluated with the deep learning model which is trained with feature representations of children faces. By discarding the softmax of the VGG-Face model and extracting CNN image features to train a multi class SVM, it was possible to achieve superior performance. Performance of the proposed system is tested using the photographs of children with different lighting conditions, noises and also images at different ages of children. The classification achieved a higher accuracy of 99.41% which shows that the proposed methodology of face recognition could be used for reliable missing children identification.

**REFERENCES**

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", Nature, 521(7553):436–444, 2015.

[2] O. Deniz, G. Bueno, J. Salido, and F. D. la Torre, "Face recognition using histograms of oriented gradients", Pattern Recognition Letters, 32(12):1598–1603, 2011.

[3] C. Geng and X. Jiang, "Face recognition using sift features", IEEE International Conference on Image Processing(ICIP), 2009.

[4] Rohit Satle, Vishnuprasad Poojary, John Abraham, Shilpa Wakode, "Missing child identification using face recognition system", International Journal of Advanced Engineering and Innovative Technology (IJAEIT), Volume 3 Issue 1 July - August 2016.

[5] <https://en.wikipedia.org/wiki/FindFace>

[6] <https://www.reuters.com/article/us-china-trafficking-apps/mobileapp-helps-china-recover-hundreds-of-missing-childrenidUSKBN15J0GU>

[7] Simonyan, Karen and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition", International Conference on Learning Representations ( ICLR), April 2015. [8] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep Face Recognition," in British Machine Vision Conference, vol. 1, no. 3, pp. 1-12, 2015.

[9] A. Vedaldi, and K. Lenc, "MatConvNet: Convolutional Neural Networks for MATLAB", ACM International Conference on Multimedia, Brisbane, October 2015