Annexure- I

Machine Learning Foundation

Project Name: AGE AND GENDER DETECTION

A Project report

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology

(Computer Science and Engineering)

Submitted to

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB

Date: 10th April 2022

SUBMITTED BY

Name of Student: P Ajay Kumar, K Vinay Reddy

Registration Number:11911903, 11901380

Annexure-II: Student Declaration

To whom so ever it may concern

I, **P. Ajay Kumar 11911903, K. Vinay Reddy 11901380, we hear by** declare that the work done by me on "**AGE AND GENDER DETECTION** " from February, 2022 to April, 2022 is a record of orginal work for the partial fulfilment of the requirements for the award of the degree ,

Bachelor of Technology.

Name of the Student:

P Ajay Kumar, 11911903, K Vinay Reddy 11901380

Date of submission : 13th April 2022

Acknowledgement

I would like to thank my teacher UPINDER KAUR who gave me this opportunity to work on this project. I got to learn a lot from this project about Age and gender detection.

At last, I would like to extend my heartfelt thanks to my parents because without their help this project would not have been successful. Finally, I would like to thank my dear friends who have been with me all the time.

Table of Contents

Introduction

Facial analysis has gained much recognition in the computer vision community in the recent past . Human's face contains features that determine identity, age, gender, emotions, and the ethnicity of people . Among these features, age and gender classification can be especially helpful in several real-world applications including security and video surveillance, electronic customer relationship management, biometrics, electronic vending machines, human-computer interaction, entertainment, cosmetology, and forensic art. However, several issues in age and gender classification are still open problems. Age and gender predictions of unfiltered real-life faces are yet to meet the requirements of commercial and real-world applications in spite of the progress computer vision community keeps making with the continuous improvement of the new techniques that improve the state of the art .Over the past years, a lot of methods have been proposed to solve the classifications problem. Many of those methods are handcrafted which perform unsatisfactorily on the age and gender predictions of unconstrained in-the-wild images . These conventional hand-engineered methods relied on the differences in dimensions of facial features and face descriptors which do not have the ability to handle the varying degrees of variation observed in these challenging unconstrained imaging conditions. ,e images in these categories have some variations in appearance, noise, pose, and lighting which may affect the ability of those manually designed computer vision methods to accurately classify the age and gender of the images. Recently, deep learning-based methods have shown encouraging performance in this field especially on the age and gender classification of unfiltered face images. In light of the current works in age and gender classification and encouraging signs of progress in deep learning and CNN, we therefore propose a novel end-to-end deep learning-based classification model that predicts age group and gender of unfiltered in-the-wild facial images. We formulate the age and gender classifications task as a classification problem in which the CNN model learns to predict the age and gender from a face image.

The contributions of this work are summarized as follows:

(1) We propose a model that uses CNN architecture to predict the age group and gender of human's faces from unfiltered real-world environments. The novel CNN approach addresses the age and gender labels as a set of discrete annotations and train the classifiers that predict the human's age group and gender.

(2) We design a quality and robust image pre processing algorithm that prepare and pre process the unfiltered images for the CNN model and this greatly has a very strong impact on the performance accuracy of our age and gender classifiers.

(3) We demonstrate that pretraining on large-scale datasets allows an effective training of our age and gender CNN model which enable the classifiers to generalize on the test images and then avoid overfitting.

(4) Finally, OIU-Adience benchmark is used to evaluate the performance of our novel CNN model, and despite the very challenging nature of the images in the dataset, our approach produces significant improvements in age group and gender classification accuracy over the state-of-the-art methods; the result can satisfy the requirements of several real-world applications.

Related Works

In this section, we briefly review the age and gender classification literature and describe both the early methods and those that are most related to our proposed method, focussing on age and gender classification of face images from unconstrained real-world environments. Almost all of the early methods in age and gender classifications were handcrafted, focussing on manually engineering the facial features from the face and mainly provides a study on constrained images that were taken from controlled imaging conditions. To mention a few, in 1999, Kwon and Lobo developed the very first method for age estimation focussing on geometric features of the face that determine the ratios among different dimensions of facial features. These geometric features separate babies from adult successfully but are incapable of distinguishing between young adult and senior adult. Hence, in 2004, Lanitis et al. proposed an Active Appearance Model (AAM) based method that included both the geometric and texture features, for the estimation task. ,is method is not suitable for the unconstrained imaging conditions attributed to real-world face images which have different degrees of variations in illumination, expression, poses, and so forth. From 2007, most of the approaches also employed manually designed features for the estimation task: Gabor, Spatially Flexible Patches (SFP), Local Binary Patterns (LBP), and Biologically Inspired Features (BIF) . In recent years ,classification

and regression methods are employed to classify the age and gender of facial images using those features. Classification methods used Support Vector Machine (SVM) based methods for age and gender classification. Linear regression , Support Vector Regression (SVR), Canonical Correlation Analysis (CCA), and Partial Least Squares (PLS) are the common regression methods for age and gender predictions. Dileep and Danti also proposed an approach that used feed-forward propagation neural networks and 3-sigma control limits approach to classify people's age into children, middle-aged adults, and old-aged adults. However, all of these methods are only suitable and effective on constrained imaging conditions; they cannot handle the unconstrained nature of the real-world images and, therefore, cannot be relied on to achieve respectable performance on the in-the wild images which are common in practical applications.

More recently, an expanding number of researchers start to use CNN for age and gender classification. It can classify the age and gender of unfiltered face images relying on its good feature extraction technique . Availability of sufficiently large data for training and high end computer machines also help in the adoption of the deep CNN methods for the classification task. CNN model can learn compact and discriminative facial features, especially when the volume of training images is sufficiently large, to obtain the relevant information needed for the two classifications. For example, in 2015, Levi et al. proposed a CNN based model, comprising of five layers, three convolutional and two fully connected layers, to predict the age of real-world face images. ,e model included centre-crop and oversampling method, to handle the small misalignment in unconstrained images. Yi et al. , in their paper, applied an end-to-end multitask CNN system that learns a deeper structure and the parameters needed, to solve the age, gender, and ethnicity classification task. In, the authors investigated a pretrained deep VGG-Face CNN approach, for automatic age estimation from real-world face images. The CNN based model consists of eleven layers, including eight convolutional and three fully connected layers. They also proposed a novel CNN based method, forage group and gender estimation: Residual Networks of Residual Networks (RoR). ,e model includes an RoR architecture, which was pretrained on gender and weighted loss layer and then on ImageNet dataset, and finally it was fine-tuned on IMDb-WIKI-101 dataset. Ranjan et al. In presented a model that simultaneously solves a set of face analysis tasks, using a single CNN. ,e end-to-end solution is a novel multitask learning CNN framework, which shares the parameters from lower layers of CNN among all the tasks for gender recognition, age estimation, etc. In, the authors proposed a CNN solution for age estimation, from a single face image. ,e CNN based solution includes a robust face alignment phase that prepares and preprocesses the face images before being fed to the designed model. The authors also collected large-scale face images, with age and gender

label: IMDb-WIKI dataset. In 2018, Liu et al. Developed a CNN based model that employed a multiclass focal loss function. ,e age estimation model was validated on Adience benchmark for performance accuracy, and it achieved a comparable result with state-of-the-art methods. Also in, Duan et al. introduced a hybrid CNN structure for age and gender classification. ,e model includes a CNN and Extreme Learning Machine (ELM). The CNN extracts the features from the input images while ELM classifies the intermediate results. In, the authors proposed a robust estimations solution (CNN2ELM) that also includes a CNN and ELM. ,e model, an improvement of the work in, is three CNN based solutions for age, gender, and race classification from face images. The authors in also proposed a novel method based on "attention long short-term memory (LSTM) network" for age estimation in-the-wild .The method was evaluated on Adience, MORPH-II, FGNET, LAP15, and LAP16 datasets for performance evaluation. Also in , the authors studied an age group-n encoding CNN based model: AGE. The model explores the relationship between the real age and its adjacent ages, by grouping adjacent ages into the same group.

Unfortunately, some of these methods mentioned above have been verified effectively on constrained imaging conditions; few studied the unconstrained imaging conditions. However, it is still a challenging problem classifying unconstrained faces with large variations in illumination, viewpoint, nonfrontal, etc. ,ere is a need for a suitable and robust model that can improve the state-of-the-art methods for its applicability in intelligent and real-world applications. Here, we address those issues by designing a robust image preprocessing algorithm, pretrain the model on large-scale facial aging benchmarks with noisy age and gender labels, and also regularize the CNN parameters with our novel CNN framework.

Overview of the project

In this Python Project, we will use Deep Learning to accurately identify the gender and age of a person from a single image of a face. We will use the models trained by Tal Hassner and Gil Levi. The predicted gender may be one of 'Male' and 'Female', and the predicted age may be one of the following ranges- (0 – 2), (4 – 6), (8 – 12), (15 – 20), (25 – 32), (38 – 43), (48 – 53), (60 – 100) (8 nodes in the final softmax layer). It is very difficult to accurately guess an exact age from a single image because of factors like makeup, lighting, obstructions, and facial expressions. And so, we make this a classification problem instead of making it one of regression.

The CNN Architecture

The convolutional neural network for this python project has 3 convolutional layers:

- Convolutional layer; 96 nodes, kernel size 7
- Convolutional layer; 256 nodes, kernel size 5
- Convolutional layer; 384 nodes, kernel size 3

It has 2 fully connected layers, each with 512 nodes, and a final output layer of softmax type.

To go about the python project, we'll:

- Detect faces
- Classify into Male/Female
- Classify into one of the 8 age ranges
- Put the results on the image and display it

The Dataset

For this python project, we'll use the Adience dataset; the dataset is available in the public domain . This dataset serves as a benchmark for face photos and is inclusive of various real-world imaging conditions like noise, lighting, pose, and appearance. The images have been collected from Flickr albums and distributed under the Creative Commons (CC) license. It has a total of 26,580 photos of 2,284 subjects in eight age ranges (as mentioned above) and is about 1GB in size. The models we will use have been trained on this dataset.

Steps of the project:

1.For face detection, we have a .pb file- this is a protobuf file (protocol buffer); it holds the graph definition and the trained weights of the model. We can use this to run the trained model. And while a .pb file holds the protobuf in binary format, one with the .pbtxt extension holds it in text format. These are TensorFlow files. For age and gender, the .prototxt files describe the network configuration and the .caffemodel file defines the internal states of the parameters of the layers.

2. For face, age, and gender, initialize protocol buffer and model.

3. Initialize the mean values for the model and the lists of age ranges and genders to classify from.

4. Now, use the read Net() method to load the networks. The first parameter holds trained weights and the second carries network configuration.

5. Let's capture video stream in case you'd like to classify on a webcam's stream. Set padding to 20.

6. Now until any key is pressed, we read the stream and store the content into the names hasFrame and frame. If it isn't a video, it must wait, and so we call up waitKey() from cv2, then break.

7.Let's make a call to the highlightFace() function with the faceNet and frame parameters, and what this returns, we will store in the names resultImg and faceBoxes. And if we got 0 faceBoxes, it means there was no face to detect. Here, net is faceNet- this model is the DNN Face Detector and holds only about 2.7MB on disk.

- Create a shallow copy of frame and get its height and width.
- Create a blob from the shallow copy.
- Set the input and make a forward pass to the network.
- faceBoxes is an empty list now. for each value in 0 to 127, define the confidence (between 0 and 1). Wherever we find the confidence greater than the confidence threshold, which is 0.7, we get the x1, y1, x2, and y2 coordinates and append a list of those to faceBoxes.
- Then, we put up rectangles on the image for each such list of coordinates and return two things: the shallow copy and the list of faceBoxes.

8. But if there are indeed faceBoxes, for each of those, we define the face, create a 4-dimensional blob from the image. In doing this, we scale it, resize it, and pass in the mean values.

9. We feed the input and give the network a forward pass to get the confidence of the two class. Whichever is higher, that is the gender of the person in the picture.

10.Then, we do the same thing for age.

11. We'll add the gender and age texts to the resulting image and display it with imshow().

Prerequisites

You'll need to install OpenCV (cv2) to be able to run this project. You can do this with pip-

pip install opencv-python

Other packages you'll be needing are math and argparse, but those come as part of the standard Python library.

This are the required modules

- opencv_face_detector.pbtxt
- opencv_face_detector_uint8.pb
- age_deploy.prototxt
- age_net.caffemodel
- gender_deploy.prototxt
- gender_net.caffemodel

- gender_net.caffemodel: It is the pre-trained model weights for gender detection.
- deploy_gender.prototxt: is the model architecture for the gender detection model (a plain text file with a JSON-like structure containing all the neural network layer's definitions).
- age_net.caffemodel: It is the pre-trained model weights for age detection.
- deploy_age.prototxt: is the model architecture for the age detection model (a plain text file with a JSON-like structure containing all the neural network layer's definitions). G
- Opencv_face_detector_uint8.pb: The pre-trained model weights for face detection,
- Opencv_face_txt: This is the model architecture for the face detection model

Software Tools:

Computer vision:

Computer vision is an interdisciplinary scientific field that deals with how computers can gain high-level understanding from digital images or videos. From the perspective of engineering, it seeks to understand and automate tasks that the human visual system can do.

Computer vision tasks include methods for acquiring, processing, analyzing and understanding digital images, and extraction of high-dimensional data from the real world in order to produce numerical or symbolic information, e.g. in the forms of decisions. Understanding in this context means the transformation of visual images (the input of the retina) into descriptions of the world that make sense to thought processes and can elicit appropriate action. This image understanding can be seen as the disentangling of symbolic information from image data using models constructed with the aid of geometry, physics, statistics, and learning theory.
The scientific discipline of computer vision is concerned with the theory behind artificial systems that extract information from images. The image data can take many forms, such as video sequences, views from multiple cameras, multi-dimensional data from a 3D scanner, or medical scanning device. The technological discipline of computer vision seeks to apply its theories and models to the construction of computer vision systems.
Sub-domains of computer vision include scene reconstruction, object detection, event detection, video tracking, object recognition, 3D pose estimation, learning, indexing, motion estimation, visual servoing, 3D scene modeling, and image restoration.

History

In the late 1960s, computer vision began at universities which were pioneering artificial intelligence. It was meant to mimic the human visual system, as a stepping stone to endowing robots with intelligent behavior. In 1966, it was believed that this could be achieved through a summer project, by attaching a camera to a computer and having it "describe what it saw".
What distinguished computer vision from the prevalent field of digital image processing at that time was a desire to extract three-dimensional structure from images with the goal of achieving full scene understanding. Studies in the 1970s formed the early foundations for many of the computer vision algorithms that exist today, including extraction of edges from images, labeling of lines, non-polyhedral and polyhedral modeling, representation of objects as interconnections of smaller structures, optical flow, and motion estimation

The next decade saw studies based on more rigorous mathematical analysis and quantitative aspects of computer vision. These include the concept of scale-space, the inference of shape from various cues such as shading, texture and focus, and contour models known as snakes. Researchers also realized that many of these mathematical concepts could be treated within the same optimization framework as regularization and Markov random fields. By the 1990s, some of the previous research topics became more active than the others. Research in projective 3-D reconstructions led to better understanding of camera calibration. With the advent of optimization methods for camera calibration, it was realized that a lot of the ideas were already explored in bundle adjustment theory from the field of photogrammetry. This led to methods for sparse 3-D reconstructions of scenes from multiple images. Progress was made on the dense stereo correspondence problem and further multi-view stereo techniques. At the same time, variations of graph cut were used to solve image segmentation. This decade also marked the first time statistical learning techniques were used in practice to recognize faces in images (see Eigenface). Toward the end of the 1990s, a significant change came about with the increased interaction between the fields of computer graphics and computer vision. This included image-based rendering, image morphing, view interpolation, panoramic image stitching and early light-field rendering. Recent work has seen the resurgence of feature-based methods, used in conjunction with machine learning techniques and complex optimization frameworks. The advancement of Deep Learning techniques has brought further life to the field of computer vision. The accuracy of deep learning algorithms on several benchmark computer vision data sets for tasks ranging from classification, segmentation and optical flow has surpassed prior methods.

Applications

Applications range from tasks such as industrial machine vision systems which, say, inspect bottles speeding by on a production line, to research into artificial intelligence and computers or robots that can comprehend the world around them. The computer vision and machine vision fields have significant overlap. Computer vision covers the core technology of automated image analysis which is used in many fields. Machine vision usually refers to a process of combining automated image analysis with other methods and technologies to provide automated inspection and robot guidance in industrial applications. In many computer-vision applications, the computers are pre-programmed to solve a particular task, but methods based on learning are now becoming increasingly common. Examples of applications of computer vision include systems for:

Automatic inspection, *e.g.*, in manufacturing applications;

- Assisting humans in identification tasks, e.g., a species identification system;
- Controlling processes, *e.g.*, an industrial robot;
- Detecting events, *e.g.*, for visual surveillance or people counting, e.g., in the restaurant industry;
- Interaction, *e.g.*, as the input to a device for computer-human interaction;
- Modeling objects or environments, *e.g.*, medical image analysis or topographical modeling;
- Navigation, *e.g.*, by an autonomous vehicle or mobile robot; and
- Organizing information, *e.g.*, for indexing databases of images and image sequences.
- Tracking surfaces or planes in 3D coordinates for allowing Augmented Reality experiences.

Medicine

One of the most prominent application fields is medical computer vision, or medical image processing, characterized by the extraction of information from image data to diagnose a patient. An example of this is detection of tumours, arteriosclerosis or other malign changes; measurements of organ dimensions, blood flow, etc. are another example. It also supports medical research by providing new information: *e.g.*, about the structure of the brain, or about the quality of medical treatments. Applications of computer vision in the medical area also includes enhancement of images interpreted by humans—ultrasonic images or X-ray images for example—to reduce the influence of noise.

Military

Military applications are probably one of the largest areas for computer vision. The obvious examples are detection of enemy soldiers or vehicles and missile guidance. More advanced systems for missile guidance send the missile to an area rather than a specific target, and target selection is made when the missile reaches the area based on locally acquired image data. Modern military concepts, such as "battlefield awareness", imply that various sensors, including image sensors, provide a rich set of information about a combat scene which

can be used to support strategic decisions. In this case, automatic processing of the data is used to reduce complexity and to fuse information from multiple sensors to increase reliability.

Autonomous vehicles

One of the newer application areas is autonomous vehicles, which include submersibles, land-based vehicles (small robots with wheels, cars or trucks), aerial vehicles, and unmanned aerial vehicles (UAV). The level of autonomy ranges from fully autonomous (unmanned) vehicles to vehicles where computer-vision-based systems support a driver or a pilot in various situations. Fully autonomous vehicles typically use computer vision for navigation, *e.g.* for knowing where it is, or for producing a map of its environment (SLAM) and for detecting obstacles. It can also be used for detecting certain task-specific events, *e.g.*, a UAV looking for forest fires. Examples of supporting systems are obstacle warning systems in cars and systems for autonomous landing of aircraft. Several car manufacturers have demonstrated systems for autonomous driving of cars, but this technology has still not reached a level where it can be put on the market. There are ample examples of military autonomous vehicles ranging from advanced missiles to UAVs for recon missions or missile guidance. Space exploration is already being made with autonomous vehicles using computer vision, *e.g.*, NASA's *Curiosity* and CNSA's *Yutu-2* rover.

Tactile Feedback

Materials such as rubber and silicon are being used to create sensors that allow for applications such as detecting micro undulations and calibrating robotic hands. Rubber can be used in order to create a mold that can be placed over a finger, inside of this mold would be multiple strain gauges. The finger mold and sensors could then be placed on top of a small sheet of rubber containing an array of rubber pins. A user can then wear the finger mold and trace a surface. A computer can then read the data from the strain gauges and measure if one or more of the pins is being pushed upward. If a pin is being pushed upward then the computer can recognize this as an imperfection in the surface. This sort of technology is useful in order to receive accurate data of the imperfections on a very large surface. Another variation of this finger mold sensor are sensors that contain a camera suspended in silicon. The silicon forms a dome around the

outside of the camera and embedded in the silicon are point markers that are equally spaced. These cameras can then be placed on devices such as robotic hands in order to allow the computer to receive highly accurate tactile data.

Other application areas include:

- Support of visual effects creation for cinema and broadcast, *e.g.*, camera tracking (matchmoving).
- Surveillance.
- Driver drowsiness detection
- Tracking and counting organisms in the biological sciences

System methods

The organization of a computer vision system is highly application-dependent. Some systems are stand-alone applications that solve a specific measurement or detection problem, while others constitute a sub-system of a larger design which, for example, also contains sub-systems for control of mechanical actuators, planning, information databases, man-machine interfaces, etc. The specific implementation of a computer vision system also depends on whether its functionality is pre-specified or if some part of it can be learned or modified during operation. Many functions are unique to the application. There are, however, typical functions that are found in many computer vision systems.

- **Image acquisition** – A digital image is produced by one or several image sensors, which, besides various types of light-sensitive cameras, include range sensors, tomography devices, radar, ultra-sonic cameras, etc. Depending on the type of sensor, the resulting image data is an ordinary 2D image, a 3D volume, or an image sequence. The pixel values typically correspond to light intensity in one or several spectral bands (gray images or colour images), but can also be related to various physical measures, such as depth, absorption or reflectance of sonic or electromagnetic waves, or nuclear magnetic resonance.
- **Pre-processing** – Before a computer vision method can be applied to image data in order to extract some specific piece of information, it is usually necessary to process the data in order to assure that it satisfies certain assumptions implied by the method. Examples are:

    - Re-sampling to assure that the image coordinate system is correct.
    - Noise reduction to assure that sensor noise does not introduce false information.
    - Contrast enhancement to assure that relevant information can be detected.
    - Scale space representation to enhance image structures at locally appropriate scales.
- **Feature extraction** – Image features at various levels of complexity are extracted from the image data. Typical examples of such features are:
    - Lines, edges and ridges.
    - Localized interest points such as corners, blobs or points.

  More complex features may be related to texture, shape or motion.
- **Detection/segmentation** – At some point in the processing a decision is made about which image points or regions of the image are relevant for further processing. Examples are:
    - Selection of a specific set of interest points.
    - Segmentation of one or multiple image regions that contain a specific object of interest.
    - Segmentation of image into nested scene architecture comprising foreground, object groups, single objects or salient object parts (also referred to as spatial-taxon scene hierarchy, while the visual salience is often implemented as spatial and temporal attention.
    - Segmentation or co-segmentation of one or multiple videos into a series of per-frame foreground masks, while maintaining its temporal semantic continuity.
- **High-level processing** – At this step the input is typically a small set of data, for example a set of points or an image region which is assumed to contain a specific object. The remaining processing deals with, for example:
    - Verification that the data satisfy model-based and application-specific assumptions.
    - Estimation of application-specific parameters, such as object pose or object size.
    - Image recognition – classifying a detected object into different categories.
    - Image registration – comparing and combining two different views of the same object.

- **Decision making** Making the final decision required for the application, for example:
    - Pass/fail on automatic inspection applications.
    - Match/no-match in recognition applications.
    - Flag for further human review in medical, military, security and recognition applications.

Image-understanding systems

Image-understanding systems (IUS) include three levels of abstraction as follows: low level includes image primitives such as edges, texture elements, or regions; intermediate level includes boundaries, surfaces and volumes; and high level includes objects, scenes, or events. Many of these requirements are entirely topics for further research.

The representational requirements in the designing of IUS for these levels are: representation of prototypical concepts, concept organization, spatial knowledge, temporal knowledge, scaling, and description by comparison and differentiation.

While inference refers to the process of deriving new, not explicitly represented facts from currently known facts, control refers to the process that selects which of the many inference, search, and matching techniques should be applied at a particular stage of processing. Inference and control requirements for IUS are: search and hypothesis activation, matching and hypothesis testing, generation and use of expectations, change and focus of attention, certainty and strength of belief, inference and goal satisfaction.

Opencv

OpenCV (*Open Source Computer Vision Library*) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel The library is cross-platform and free for use under the open-source . Starting with 2011, OpenCV features GPU acceleration for real-time operations

History

Officially launched in 1999 the OpenCV project was initially an Intel Research initiative to advance CPU-intensive applications, part of a series of projects including real-time ray tracing and 3D display walls. The main contributors to the project included a number of optimization experts in Intel Russia, as well as Intel's Performance Library Team. In the early days of OpenCV, the goals of the project were described as:

- Advance vision research by providing not only open but also optimized code for basic vision infrastructure. No more reinventing the wheel.
- Disseminate vision knowledge by providing a common infrastructure that developers could build on, so that code would be more readily readable and transferable.
- Advance vision-based commercial applications by making portable, performance-optimized code available for free – with a license that did not require code to be open or free itself.

The first alpha version of OpenCV was released to the public at the IEEE Conference on Computer Vision and Pattern Recognition in 2000, and five betas were released between 2001 and 2005. The first 1.0 version was released in 2006. A version 1.1 "pre-release" was released in October 2008.

The second major release of the OpenCV was in October 2009. OpenCV 2 includes major changes to the C++ interface, aiming at easier, more type-safe patterns, new functions, and better implementations for existing ones in terms of performance (especially on multi-core systems). Official releases now occur every six months and development is now done by an independent Russian team supported by commercial corporations.
In August 2012, support for OpenCV was taken over by a non-profit foundation OpenCV.org, which maintains a developer and user site.
In May 2016, Intel signed an agreement to acquire Itseez, a leading developer of OpenCV,
In July 2020, OpenCV announced and began a Kickstarter campaign for the OpenCV AI Kit, a series of hardware modules and additions to OpenCV supporting Spatial AI.

Applications

OpenCV's application areas include:

- 2D and 3D feature toolkits
- Egomotion estimation
- Facial recognition system
- Gesture recognition
- Human–computer interaction (HCI)
- Mobile robotics
- Motion understanding
- Object detection
- Segmentation and recognition
- Stereopsis stereo vision: depth perception from 2 cameras
- Structure from motion (SFM)
- Motion tracking
- Augmented reality

To support some of the above areas, OpenCV includes a statistical machine learning library that contains:

- Boosting
- Decision tree learning

- Gradient boosting trees
- Expectation-maximization algorithm
- k-nearest neighbor algorithm
- Naive Bayes classifier
- Artificial neural networks
- Random forest
- Support vector machine (SVM)
- Deep neural networks (DNN)

Convolutional neural network

In deep learning, a **convolutional neural network** (**CNN**, or **ConvNet**) is a class of Artificial Neural Network (**ANN**), most commonly applied to analyze visual imagery. They are also known as **Shift Invariant** or **Space Invariant Artificial Neural Networks** (**SIANN**), based on the shared-weight architecture of the convolution kernels or filters that slide along input features and provide translation equivariant responses known as feature maps. Counter-intuitively, most convolutional neural networks are only equivariant, as opposed to invariant, to translation. They have applications in image and video recognition, recommender systems, image classification, image segmentation, medical image analysis, natural language processing, brain-computer interfaces, and financial time series.
CNNs are regularized versions of multilayer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "full connectivity" of these networks make them prone to overfitting data. Typical ways of regularization, or preventing overfitting, include: penalizing parameters during training (such as weight decay) or trimming connectivity (skipped connections, dropout, etc.) CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble patterns of increasing complexity using smaller and simpler patterns embossed in their filters. Therefore, on a scale of connectivity and complexity, CNNs are on the lower extreme.

- Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field.

- CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns to optimize the filters (or kernels) through automated learning, whereas in traditional algorithms these filters are hand-engineered. This independence from prior knowledge and human intervention in feature extraction is a major advantage.

Architecture

A convolutional neural network consists of an input layer, hidden layers and an output layer. In any feed-forward neural network, any middle layers are called hidden because their inputs and outputs are masked by the activation function and final convolution. In a convolutional neural network, the hidden layers include layers that perform convolutions. Typically this includes a layer that performs a dot product of the convolution kernel with the layer's input matrix. This product is usually the Frobenius inner product, and its activation function is commonly ReLU. As the convolution kernel slides along the input matrix for the layer, the convolution operation generates a feature map, which in turn contributes to the input of the next layer. This is followed by other layers such as pooling layers, fully connected layers, and normalization layers.

Convolutional layers

In a CNN, the input is a tensor with a shape: (number of inputs) x (input height) x (input width) x (input channels). After passing through a convolutional layer, the image becomes abstracted to a feature map, also called an activation map, with shape: (number of inputs) x (feature map height) x (feature map width) x (feature map channels).
Convolutional layers convolve the input and pass its result to the next layer. This is similar to the response of a neuron in the visual cortex to a specific stimulus. Each convolutional neuron processes data only for its receptive field. Although fully connected feedforward neural networks can be used to learn features and classify data, this architecture is generally impractical for larger inputs such as high resolution images. It would require a very high number of neurons, even in a shallow architecture, due to the large input size of images, where each pixel is a relevant input feature. For instance, a fully connected layer for a (small) image of size 100 x 100 has 10,000 weights for *each* neuron in the second layer. Instead, convolution reduces the number of free parameters, allowing the network to be deeper. For example, regardless of image size, using a 5 x 5 tiling region, each with the same shared weights, requires only 25 learnable parameters. Using regularized weights over fewer parameters avoids

the vanishing gradients and exploding gradients problems seen during backpropagation in traditional neural networks. Furthermore, convolutional neural networks are ideal for data with a grid-like topology (such as images) as spatial relations between separate features are taken into account during convolution and/or pooling.

Pooling layers

Convolutional networks may include local and/or global pooling layers along with traditional convolutional layers. Pooling layers reduce the dimensions of data by combining the outputs of neuron clusters at one layer into a single neuron in the next layer. Local pooling combines small clusters, tiling sizes such as 2 x 2 are commonly used. Global pooling acts on all the neurons of the feature map. There are two common types of pooling in popular use: max and average. *Max pooling* uses the maximum value of each local cluster of neurons in the feature map, while *average pooling* takes the average value.

Fully connected layers

Fully connected layers connect every neuron in one layer to every neuron in another layer. It is the same as a traditional multi-layer perceptron neural network (MLP). The flattened matrix goes through a fully connected layer to classify the images.

Receptive field

In neural networks, each neuron receives input from some number of locations in the previous layer. In a convolutional layer, each neuron receives input from only a restricted area of the previous layer called the neuron's *receptive field*. Typically the area is a square (e.g. 5 by 5 neurons). Whereas, in a fully connected layer, the receptive field is the *entire previous layer*. Thus, in each convolutional layer, each neuron takes input from a larger area in the input than previous layers. This is due to applying the convolution over and over, which takes into account the value of a pixel, as well as its surrounding pixels. When using dilated layers, the number of pixels in the receptive field remains constant, but the field is more sparsely populated as its dimensions grow when combining the effect of several layers.

Weights

Each neuron in a neural network computes an output value by applying a specific function to the input values received from the receptive field in the previous layer. The function that is applied to the input values is determined by a vector of weights and a bias (typically real numbers). Learning consists of iteratively adjusting these biases and weights.

The vector of weights and the bias are called *filters* and represent particular features of the input (e.g., a particular shape). A distinguishing feature of CNNs is that many neurons can share the same filter. This reduces the memory footprint because a single bias and a single vector of weights are used across all receptive fields that share that filter, as opposed to each receptive field having its own bias and vector weighting.

Receptive fields in the visual cortex

Work by Hubel and Wiesel in the 1950s and 1960s showed that catp visual **cortices** contain neurons that individually respond to small regions of the visual field. Provided the eyes are not moving, the region of visual space within which visual stimuli affect the firing of a single neuron is known as its receptive field. Neighboring cells have similar and overlapping receptive fields. Receptive field size and location varies systematically across the cortex to form a complete map of visual space.The cortex in each hemisphere represents the contralateral visual field.

Their 1968 paper identified two basic visual cell types in the brain:

- simple cells, whose output is maximized by straight edges having particular orientations within their receptive field
- complex cells, which have larger receptive fields, whose output is insensitive to the exact position of the edges in the field.

Hubel and Wiesel also proposed a cascading model of these two types of cells for use in pattern recognition tasks.

Neocognitron, origin of the CNN architecture

The "neocognitron" was introduced by Kunihiko Fukushima in 1980. It was inspired by the above-mentioned work of Hubel and Wiesel. The neocognitron introduced the two basic types of layers in CNNs: convolutional layers, and downsampling layers. A convolutional layer contains units whose receptive fields cover a patch of the previous layer. The weight vector (the set of adaptive parameters) of such a unit is often called a filter. Units can share filters. Downsampling layers contain units whose receptive fields cover patches of previous convolutional layers. Such a unit typically computes the average of the activations of the units in its patch. This downsampling helps to correctly classify objects in visual scenes even when the objects are shifted.
In a variant of the neocognitron called the cresceptron, instead of using Fukushima's spatial averaging, J. Weng et al. introduced a method called max-pooling where a downsampling unit computes the maximum of the activations of the units in its patch. Max-pooling is often used in modern CNNs.
Several supervised and unsupervised learning algorithms have been proposed over the decades to train the weights of a neocognitron. Today, however, the CNN architecture is usually trained through backpropagation.
The neocognitron is the first CNN which requires units located at multiple network positions to have shared weights.
Convolutional neural networks were presented at the Neural Information Processing Workshop in 1987, automatically analyzing time-varying signals by replacing learned multiplication with convolution in time, and demonstrated for speech recognition.

Time delay neural networks

The time delay neural network (TDNN) was introduced in 1987 by Alex Waibel et al. and was one of the first convolutional networks, as it achieved shift invariance. It did so by utilizing weight sharing in combination with Backpropagation training. Thus, while also using a pyramidal structure as in the neocognitron, it performed a global optimization of the weights instead of a local one.

TDNNs are convolutional networks that share weights along the temporal dimension. They allow speech signals to be processed time-invariantly. In 1990

Hampshire and Waibel introduced a variant which performs a two dimensional convolution. Since these TDNNs operated on spectrograms, the resulting phoneme recognition system was invariant to both shifts in time and in frequency. This inspired translation invariance in image processing with CNNs. The tiling of neuron outputs can cover timed stages.

TDNNs now achieve the best performance in far distance speech recognition.

Max pooling

In 1990 Yamaguchi et al. introduced the concept of max pooling, which is a fixed filtering operation that calculates and propagates the maximum value of a given region. They did so by combining TDNNs with max pooling in order to realize a speaker independent isolated word recognition system. In their system they used several TDNNs per word, one for each syllable. The results of each TDNN over the input signal were combined using max pooling and the outputs of the pooling layers were then passed on to networks performing the actual word classification.

Image recognition with CNNs trained by gradient descent

A system to recognize hand-written ZIP Code numbers involved convolutions in which the kernel coefficients had been laboriously hand designed.
Yann LeCun et al. (1989) used back-propagation to learn the convolution kernel coefficients directly from images of hand-written numbers. Learning was thus fully automatic, performed better than manual coefficient design, and was suited to a broader range of image recognition problems and image types.
This approach became a foundation of modern computer vision.

LeNet-5

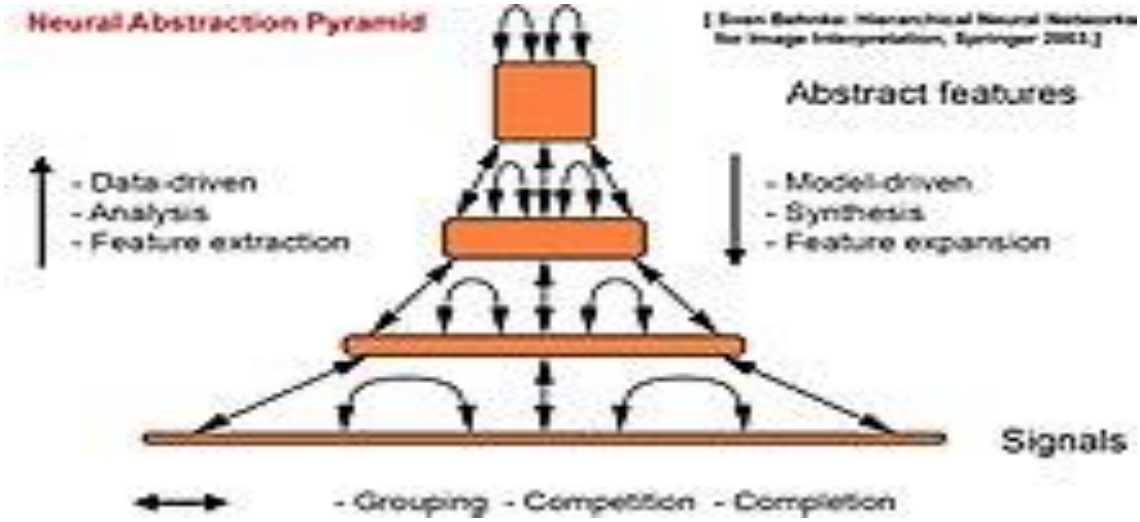
LeNet-5, a pioneering 7-level convolutional network by LeCun et al. in 1998, that classifies digits, was applied by several banks to recognize hand-written numbers on checks digitized in 32x32 pixel images. The ability to process higher resolution images requires larger and more layers of convolutional neural networks, so this technique is constrained by the availability of computing resources.

Shift-invariant neural network

Similarly, a shift invariant neural network was proposed by W. Zhang et al. for image character recognition in 1988. The architecture and training algorithm were modified in 1991 and applied for medical image processing and automatic detection of breast cancer in mammograms.
A different convolution-based design was proposed in 1988 for application to decomposition of one-dimensional electromyography convolved signals via de-convolution. This design was modified in 1989 to other de-convolution-based designs.

Neural abstraction pyramid

The feed-forward architecture of convolutional neural networks was extended in the neural abstraction pyramid by lateral and feedback connections. The



resulting recurrent convolutional network allows for the flexible incorporation of contextual information to iteratively resolve local ambiguities. In contrast to previous models, image-like outputs at the highest resolution were generated, e.g., for semantic segmentation, image reconstruction, and object localization tasks.

GPU implementations

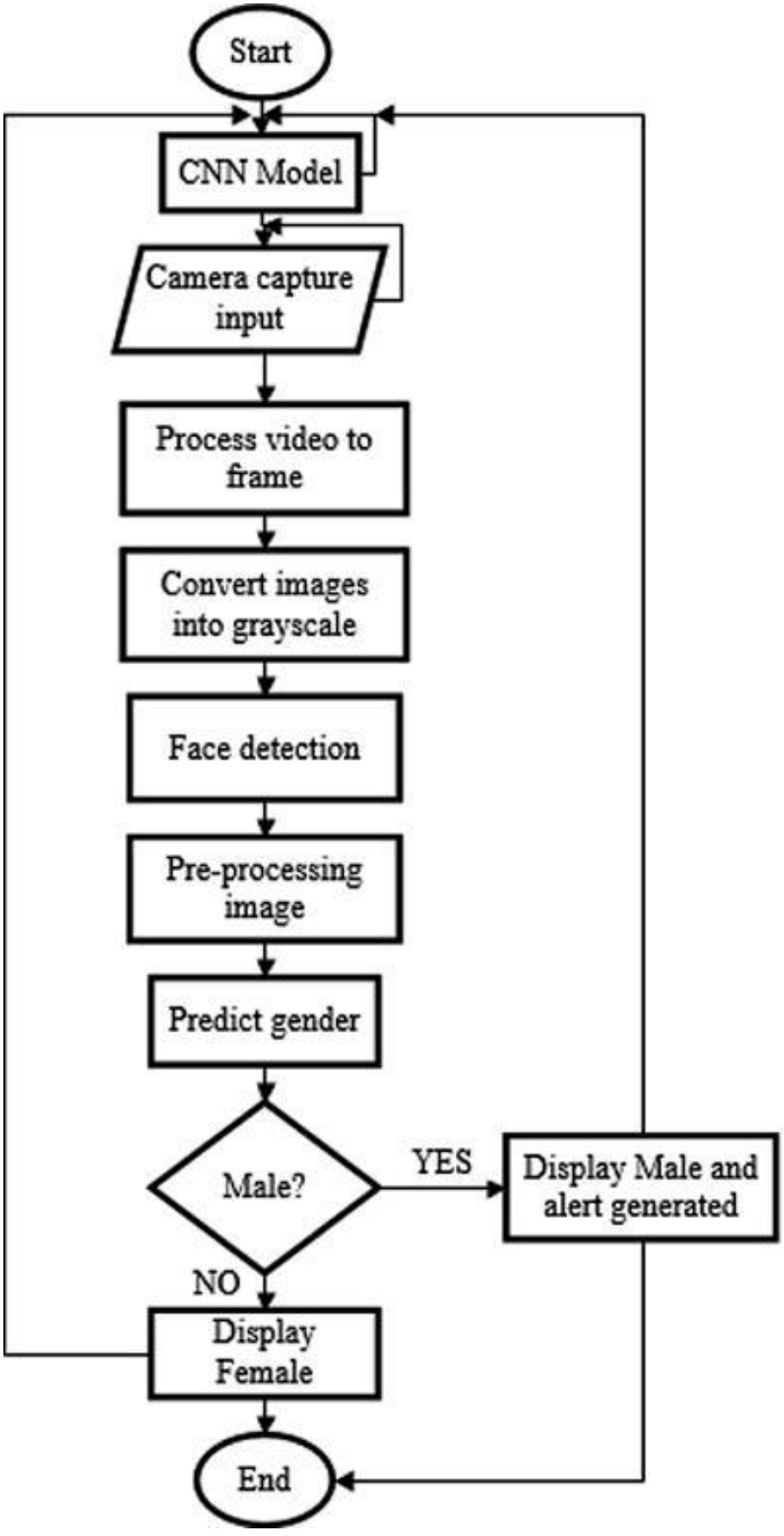
Although CNNs were invented in the 1980s, their breakthrough in the 2000s required fast implementations on graphics processing units (GPUs).

In 2004, it was shown by K. S. Oh and K. Jung that standard neural networks can be greatly accelerated on GPUs. Their implementation was 20 times faster than an equivalent implementation on CPU. In 2005, another paper also emphasised the value of GPGPU for machine learning.
The first GPU-implementation of a CNN was described in 2006 by K. Chellapilla et al. Their implementation was 4 times faster than an equivalent implementation on CPU. Subsequent work also used GPUs, initially for other types of neural networks (different from CNNs), especially unsupervised neural networks.
In 2010, Dan Ciresan et al. at IDSIA showed that even deep standard neural networks with many layers can be quickly trained on GPU by supervised learning through the old method known as backpropagation. Their network outperformed previous machine learning methods on the MNIST handwritten digits benchmark. In 2011, they extended this GPU approach to CNNs, achieving an acceleration factor of 60, with impressive results. In 2011, they used such CNNs on GPU to win an image recognition contest where they achieved superhuman performance for the first time. Between May 15, 2011 and September 30, 2012, their CNNs won no less than four image competitions. In 2012, they also significantly improved on the best performance in the literature for multiple image databases, including the MNIST database, the NORB database, the HWDB1.0 dataset (Chinese characters) and the CIFAR10 dataset (dataset of 60000 32x32 labeled RGB images). Subsequently, a similar GPU-based CNN by Alex Krizhevsky et al. won the ImageNet Large Scale Visual Recognition Challenge 2012. A very deep CNN with over 100 layers by Microsoft won the ImageNet 2015 contest.

Intel Xeon Phi implementations

Compared to the training of CNNs using GPUs, not much attention was given to the Intel Xeon Phi coprocessor. A notable development is a parallelization method for training convolutional neural networks on the Intel Xeon Phi, named Controlled Hogwild with Arbitrary Order of Synchronization (CHAOS). CHAOS exploits both the thread- and SIMD-level parallelism that is available on the Intel Xeon Phi.

CNN Process for Image using OpenCv



Applications

Image recognition

CNNs are often used in image recognition systems. In 2012 an error rate of 0.23% on the MNIST database was reported. Another paper on using CNN for image classification reported that the learning process was "surprisingly fast"; in the same paper, the best published results as of 2011 were achieved in the MNIST database and the NORB database. Subsequently, a similar CNN called AlexNet won the ImageNet Large Scale Visual Recognition Challenge 2012.

When applied to facial recognition, CNNs achieved a large decrease in error rate. Another paper reported a 97.6% recognition rate on "5,600 still images of more than 10 subjects". CNNs were used to assess video quality in an objective way after manual training; the resulting system had a very low root mean square error.

The ImageNet Large Scale Visual Recognition Challenge is a benchmark in object classification and detection, with millions of images and hundreds of object classes. In the ILSVRC 2014, a large-scale visual recognition challenge, almost every highly ranked team used CNN as their basic framework. The winner GoogLeNet (the foundation of DeepDream) increased the mean average precision of object detection to 0.439329, and reduced classification error to 0.06656, the best result to date. Its network applied more than 30 layers. That performance of convolutional neural networks on the ImageNet tests was close to that of humans. The best algorithms still struggle with objects that are small or thin, such as a small ant on a stem of a flower or a person holding a quill in their hand. They also have trouble with images that have been distorted with filters, an increasingly common phenomenon with modern digital cameras. By contrast, those kinds of images rarely trouble humans. Humans, however, tend to have trouble with other issues. For example, they are not good at classifying objects into fine-grained categories such as the particular breed of dog or species of bird, whereas convolutional neural networks handle this.

In 2015 a many-layered CNN demonstrated the ability to spot faces from a wide range of angles, including upside down, even when partially occluded, with competitive performance. The network was trained on a database of 200,000 images that included faces at various angles and orientations and a further 20 million images without faces. They used batches of 128 images over 50,000 iterations.

Video analysis

Compared to image data domains, there is relatively little work on applying CNNs to video classification. Video is more complex than images since it has another (temporal) dimension. However, some extensions of CNNs into the video domain have been explored. One approach is to treat space and time as equivalent dimensions of the input and perform convolutions in both time and space. Another way is to fuse the features of two convolutional neural networks, one for the spatial and one for the temporal stream. Long short-term memory (LSTM) recurrent units are typically incorporated after the CNN to account for inter-frame or inter-clip dependencies. Unsupervised learning schemes for training spatio-temporal features have been introduced, based on Convolutional Gated Restricted Boltzmann Machines and Independent Subspace Analysis.

Natural language processing

CNNs have also been explored for natural language processing. CNN models are effective for various NLP problems and achieved excellent results in semantic parsing, search query retrieval, sentence modeling, classification, prediction and other traditional NLP tasks. Compared to traditional language processing methods such as recurrent neural networks, CNNs can represent different contextual realities of language that do not rely on a series-sequence assumption, while RNNs are better suitable when classical time serie modeling is required

Anomaly Detection

A CNN with 1-D convolutions was used on time series in the frequency domain (spectral residual) by an unsupervised model to detect anomalies in the time domain.

Drug discovery

CNNs have been used in drug discovery. Predicting the interaction between molecules and biological proteins can identify potential treatments. In 2015, Atomwise introduced AtomNet, the first deep learning neural network for structure-based rational drug design. The system trains directly on 3-dimensional representations of chemical interactions. Similar to how image recognition networks learn to compose smaller, spatially proximate features into larger, complex structures, AtomNet discovers chemical features, such as aromaticity, sp^3 carbons and hydrogen bonding. Subsequently, AtomNet was used to predict novel candidate biomolecules for multiple disease targets, most notably treatments for the Ebola virus and multiple sclerosis.

Health risk assessment and biomarkers of aging discovery

CNNs can be naturally tailored to analyze a sufficiently large collection of time series data representing one-week-long human physical activity streams augmented by the rich clinical data (including the death register, as provided by, e.g., the NHANES study). A simple CNN was combined with Cox-Gompertz proportional hazards model and used to produce a proof-of-concept example of digital biomarkers of aging in the form of all-causes-mortality predictor.

Checkers game

CNNs have been used in the game of checkers. From 1999 to 2001, Fogel and Chellapilla published papers showing how a convolutional neural network could learn to play **checker** using co-evolution. The learning process did not use prior human professional games, but rather focused on a minimal set of information contained in the checkerboard: the location and type of pieces, and the difference in number of pieces between the two sides. Ultimately, the program (Blondie24) was tested on 165 games against players and ranked in the highest 0.4%. It also earned a win against the program Chinook at its "expert" level of play.

Go

CNNs have been used in computer Go. In December 2014, Clark and Storkey published a paper showing that a CNN trained by supervised learning from a database of human professional games could outperform GNU Go and win some games against Monte Carlo tree search Fuego 1.1 in a fraction of the time it took Fuego to play. Later it was announced that a large 12-layer convolutional neural network had correctly predicted the professional move in 55% of positions, equalling the accuracy of a 6 dan human player. When the trained convolutional network was used directly to play games of Go, without any search, it beat the traditional search program GNU Go in 97% of games, and matched the performance of the Monte Carlo tree search program Fuego simulating ten thousand playouts (about a million positions) per move.
A couple of CNNs for choosing moves to try ("policy network") and evaluating positions ("value network") driving MCTS were used by AlphaGo, the first to beat the best human player at the time.

Time series forecasting

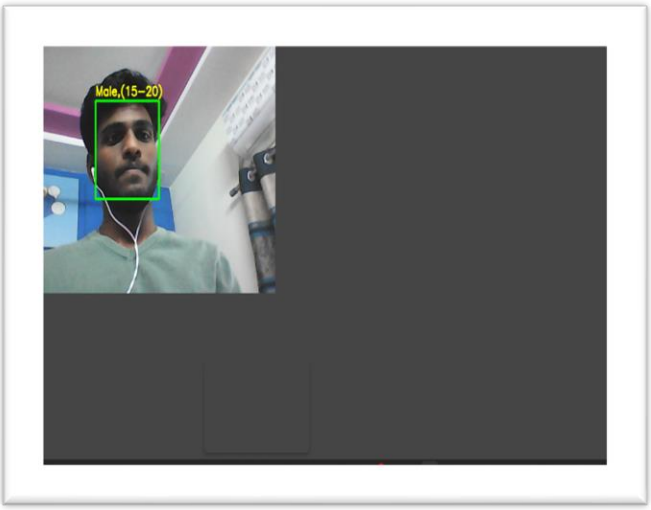
Recurrent neural networks are generally considered the best neural network architectures for time series forecasting (and sequence modeling in general), but recent studies show that convolutional networks can perform comparably or even better. Dilated convolutions might enable one-dimensional convolutional neural networks to effectively learn time series dependences. Convolutions can be implemented more efficiently than RNN-based solutions, and they do not suffer from vanishing (or exploding) gradients. Convolutional networks can provide an improved forecasting performance when there are multiple similar time series to learn from. CNNs can also be applied to further tasks in time series analysis (e.g., time series classification or quantile forecasting.

Cultural Heritage and 3D-datasets

As archaeological findings like clay tablets with cuneiform writing are increasingly acquired using 3D scanners first benchmark datasets are becoming available like *HeiCuBeDa* providing almost 2.000 normalized 2D- and 3D-datasets prepared with the GigaMesh Software Framework. So curvature based measures are used in conjunction with Geometric Neural Networks (GNNs) e.g.

for period classification of those clay tablets being among the oldest documents of human history.

Result :



Future Scope

In the future, the specified methodologies can be further improved by incorporating the below mentioned specifications in the implementation of the proposed mechanisms.

- The Gender Classification and Age Detection algorithms can be implemented with an increased number of facial image data set. This will increase the accuracy level of the output.
- The Proposed Models have been trained and tested on data sets using Linear Kernel. However the similar evaluation can be performed using other kernels of SVM Classifier like ‘rbf kernel’, ‘quadratic kernel’, etc.
- Gender Classification is performed based on the extracted feature’ lip’. The same algorithm is can be implemented on other feature(s) like eyes, nose or combination of more than one feature in human facial image.

Conclusion

It has been observed and realized that the nature, behaviour and social interaction of people is greatly dominated by his/ her gender. Therefore an efficient gender recognition and classification system would play a pivotal role in enhancing the interaction between human and the machine .Moreover, there are several other applications where gender recognition plays a crucial role which includes biometric authentication, high technology surveillance and security systems, image retrieval, and passive demographical data collections. Identification of the gender and its classification based on the distinguishable characteristics between male and female facial image can be achieved easily by the human eye. However, machines cannot visualize this difference, hence the same task becomes difficult for the computer to accomplish. Machines need meaningful data to perform gender classification. These data are usually the facial features based on which a computer classifies a facial image into either

male or female .Gender Classification is a binary Classification problem. There exist several algorithms which have been already implemented to generate a solution to the stated problem. This study addresses the issue of gender classification and age detection of the identified gender using Support Vector Machine Classifier. Although the stated methodologies have been implemented on facial image data set and results are obtained with a level of accuracy, yet there are areas which are yet to be cultivated and where further enhancement can be achieved.

References:

1. T. Ahonen, A. Hadid, and M. Pietikainen. Face description with local binary patterns: Application to face recognition. Trans. Pattern Anal. Mach. Intell., 28(12):2037–2041, 2006.

2. S. Baluja and H. A. Rowley. Boosting sex identification performance. Int. J. Comput. Vision, 71(1):111–119, 2007.

3. A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In Int.Conf. Mach. Learning, volume 3, pages 11–18, 2003.

4. W.-L. Chao, J.-Z. Liu, and J.-J. Ding. Facial age estimation based on label-sensitive learning and age-oriented regression. Pattern Recognition, 46(3):628–641, 2013.

5. K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. arXiv preprint arXiv:1405.3531, 2014

6 J. Chen, S. Shan, C. He, G. Zhao, M. Pietikainen, X.Chen, and W. Gao. Wld: A robust local image descriptor. Trans. Pattern Anal. Mach. Intell., 32(9):1705–1720, 2010

7 S. E. Choi, Y. J. Lee, S. J. Lee, K. R. Park, and J. Kim. Age estimation using a hierarchical classifier based on global and local facial features. Pattern Recognition, 44(6):1262–1281, 2011.

8 T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In European Conf. Comput. Vision, pages 484–498. Springer, 1998.

9 C. Cortes and V. Vapnik. Support-vector networks. Machine learning, 20(3):273–297, 1995.

10 E. Eidinger, R. Enbar, and T. Hassner. Age and gender estimation of unfiltered faces. Trans. on Inform. Forensics and Security, 9(12), 2014. 1, 2, 5,

11 Y. Fu, G. Guo, and T. S. Huang. Age synthesis and estimation via faces: A survey. Trans. Pattern Anal. Mach. Intell., 32(11):1955–1976, 2010.

12 Y. Fu and T. S. Huang. Human age estimation with regression on discriminative aging manifold. Int. Conf. Multimedia, 10(4):578–584, 2008.

13 K. Fukunaga. Introduction to statistical pattern recognition. Academic press, 1991.

14 A. C. Gallagher and T. Chen. Understanding images of groups of people. In Proc. Conf. Comput. Vision Pattern Recognition, pages 256–263. IEEE, 2009.

15 F. Gao and H. Ai. Face age classification on consumer images with gabor feature and fuzzy lda method. In Advances in biometrics, pages 132–141. Springer, 2009.

16 X. Geng, Z.-H. Zhou, and K. Smith-Miles. Automatic age estimation based on facial aging patterns. Trans. Pattern Anal. Mach. Intell., 29(12):2234–2240, 2007.

17 B. A. Golomb, D. T. Lawrence, and T. J. Sejnowski. Sexnet: A neural network identifies sex from human faces. In Neural Inform. Process. Syst., pages 572–579, 1990.

18 A. Graves, A.-R. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, pages 6645–6649. IEEE, 2013.