VISVESVARAYA TECHNOLOGICALUNIVERSITY

“Jnana Sangama”, Belgaum-590018.

A project Phase-I report on

“Fashion Recommendation Systems Using CNN”

Submitted in partial fulfillment for the requirements of the VII Semester degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING

For the Academic Year
2019-2020

Name:	USN:
ANJAN M	1DB16CS021
ABHISHEK V	1DB16CS005
DHEERAJ	1DB16CS051
MANIKANTA	1DB17CS041

Under the Guidance Of
MRS. HEMALATHA M
Asst. Professor,

Dept. of CSE

Department of Computer Science and Engineering

DON BOSCO INSTITUTE OF TECHNOLOGY

Kumbalagodu, Mysore Road, Bengaluru - 560074.

2019-2020

CONTENTS

Chapter 1

INTRODUCTION

Machine learning (**ML**) is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or infeasible to develop a conventional algorithm for effectively performing the task.

Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a field of study within machine learning, and focuses on exploratory data analysis through unsupervised learning. In its application across business problems, machine learning is also referred to as predictive analytics.

Machine learning tasks are classified into several broad categories. In supervised learning, the algorithm builds a mathematical model from a set of data that contains both the inputs and the desired outputs. For example, if the task were determining whether an image contained a certain object, the training data for a supervised learning algorithm would include images with and without that object (the input), and each image would have a label (the output) designating whether it contained the object. In special cases, the input may be only partially available, or restricted to special feedback.[*clarification needed*] Semi-supervised learning algorithms develop mathematical models from incomplete training data, where a portion of the sample input doesn't have labels.

Classification algorithms and regression algorithms are types of supervised learning. Classification algorithms are used when the outputs are restricted to a limited set of values. For a classification algorithm that filters emails, the input would be an incoming email, and the output would be the name of the folder in which to file the email. For an algorithm that identifies spam emails, the output would be the prediction of either "spam" or "not spam", represented by the Boolean values true and false. Regression algorithms are named for their continuous outputs, meaning they may have any value within a range. Examples of a continuous value are the temperature, length, or price of an object.

In unsupervised learning, the algorithm builds a mathematical model from a set of data which contains only inputs and no desired output labels. Unsupervised learning algorithms are used to find structure in the data, like grouping or clustering of data points. Unsupervised learning can discover patterns in the data, and can group the inputs into categories, as in feature learning. Dimensionality reduction is the process of reducing the number of "features", or inputs, in a set of data.

Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data, and consists of a set of training examples. Each training example has one or more inputs and a desired output, also known as a supervisory signal. In the mathematical model, each training example is represented by an array or vector, sometimes called a feature vector, and the training data is represented by a matrix. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. An algorithm that improves the accuracy of its outputs or predictions over time is said to have learned to perform that task.

Unsupervised learning algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points. The algorithms therefore learn from test data that has not been labeled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. A central application of unsupervised learning is in the field of density estimation in statistics, though unsupervised learning encompasses other domains involving summarizing and explaining data features.

Deep learning is part of a broader family of machine learning methods based on artificial neural networks. Learning can be supervised, semi-supervised or unsupervised.

Deep learning architectures such as deep neural networks, deep belief networks, recurrent neural networks and convolutional neural networks have been applied to fields including computer vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation, bioinformatics, drug design, medical image analysis, material inspection and board game programs, where they have produced results comparable to and in some cases superior to human experts.

Artificial Neural Networks (ANNs) were inspired by information processing and distributed communication nodes in biological systems. ANNs have various differences from biological brains. Specifically, neural networks tend to be static and symbolic, while the biological brain of most living organisms is dynamic (plastic) and analog

convolutional neural network is a class of deep neural networks, most commonly applied to analyzing visual imagery. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics. They have applications in image and video recognition, recommender systems, image classification, medical image analysis, and natural language processing.

When programming a CNN, each convolutional layer within a neural network should have the following attributes:Input is a tensor with shape (number of images) x (image width) x (image height) x (image depth).Convolutional kernels whose width and height are hyper-parameters, and whose depth must be equal to that of the image. 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 as well as classify data, it is not practical to apply this architecture to images. A very high number of neurons would be necessary, even in a shallow (opposite of deep) architecture, due to the very large input sizes associated with images, where each pixel is a relevant variable. 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. The convolution operation brings a solution to this problem as it reduces the number of free parameters, allowing the network to be deeper with fewer parameters. For instance, regardless of image size, tiling regions of size 5 x 5, each with the same shared weights, requires only 25 learnable parameters. In this way, it resolves the vanishing or exploding gradients problem in training traditional multi-layer neural networks with many layers by using backpropagation.

We use a Convolutional Neural Network (CNN) model which takes image as an input that the consumer wants to purchase. CNN classify the image category and use the feature vector of the final fully connected layer as an input vector to feed in a similarity measure CNN model to identify the likely products in the database.

Classification discovers the cluster of the product most likely to the given input image taken by the customer. We have several classes in our dataset as a whole. For example, an image of a T-shirt will be classified as “chiffon”, “linen”, “classic”, “tribal”, “embroidery”, “graphics” and similar T-shirts belongs to the input category are recommended to the customer in Recommendation.

Chapter 2

SCOPE OF PROJECT WORK

The Handwritten Written recognition(HWR) can be extended in the following directions through its implementation:

1. Font Independent HWR: An Optical Character Recognition system could be developed by considering the multiple font style in use. Our approach is very much useful for the font independent case. Because, for font or character size, it finds the string and the strings are parsed to recognize the character. Once character is identified, the corresponding character could be ejected through an efficient editor. Efforts have been taken to develop a compatible editor.

2. HWR for Languages: All languages require development of an HWR for printed characters, and for handwritten characters, HWR has to be developed for all languages. Of course, HWR for printed characters are easy when compared to the handwritten cursive scripts. Even for the printed document recognition, an OCR should be able to perform the all features besides character recognition, such as spell check, sentence and grammar check, also an editor with key board encoding and font encoding is required. With this approach the printed and handwritten characters are recognizable easily with less effort and more accuracy. A module for Skew correction and line separation, word and character separation along with an editor with spell checker and grammar checker could be designed for developing a complete HWR. Further, with a little fine tuning on the modules, such as, skew correction and line separation, word and character separation, a complete HWR could be designed for handwritten scripts of any language for that matter. It is proposed to apply the approach to all manuscripts recognition of South-Indian languages. Since some of the characters in some of the languages have similar characters viz, Tamil and Malayalam have similar features among few characters, and Telugu and Kannada have similarity among most of the characters, our approach could be applied for these languages and could be extended to all other languages.

3. Cursive Characters HWR: There is heavy demand for an HWR system which recognizes cursive scripts and manuscripts. This actually avoids keyboard typing and font encoding too.

4. Language Converter through HWR: Once a complete HWR has been developed for two languages with font encoding, spell checker and grammatical sentence check, then a converter could be implemented to convert sentences from one language to another through a

transliteration and translation scheme.

5. A Bilingual or multi-lingual script HWR: It is basically necessary to develop an HWR for multi lingual script for a country where more than 10 languages are in use officially. For example, if there is a document where two language scripts are available, then one need not separate two scripts into two different files and feed them into two HWR's. Using our approach one could develop an HWR for two languages, Tamil and English in the same document. These scripts could be edited later with an editor too.

6. Speech recognition from HWR: The most required application today is Speech recognition. The recognized Printed or Handwritten character could be recorded and through a voice synthesizer speech output could be generated. This would help the blind to send and receive information.

Chapter 3

LITERATURE SURVEY

An image retrieval system is a computer system for browsing, searching and retrieving images from a large database of digital images. Most traditional and common methods of image retrieval utilize some method of adding metadata such as captioning, keywords, title or descriptions to the images so that retrieval can be performed over the annotation words. Manual image annotation is time-consuming, laborious and expensive; to address this, there has been a large amount of research done on automatic image annotation. Additionally, the increase in social web applications and the semantic web have inspired the development of several web-based image annotation tools.

Survey of recommender systems techniques:

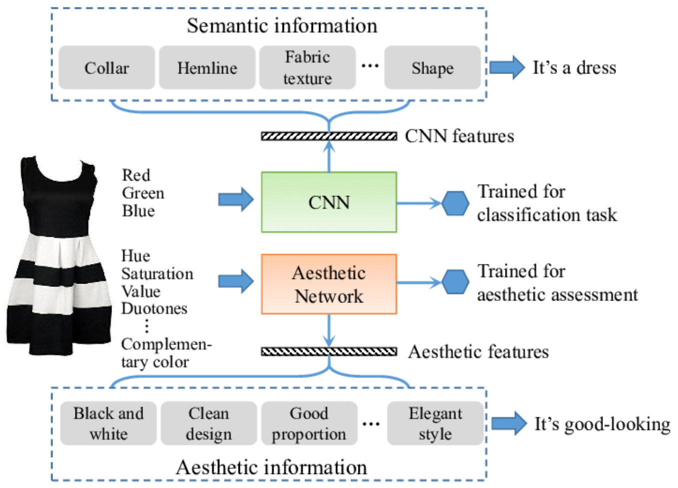


Fig: Flowchart the processing of recommendation systems

Image search is a specialized data search used to find images. To search for images, a user may provide query terms such as keyword, image file/link, or click on some image, and the system will return images "similar" to the query. The similarity used for search criteria could be meta tags, color distribution in images, region/shape attributes, etc.

Image meta search - search of images based on associated metadata such as keywords, text, etc.

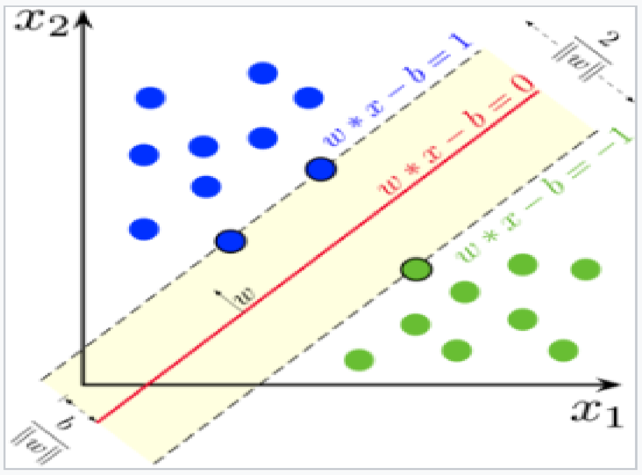
Content-based image retrieval (CBIR) – the application of computer vision to the image retrieval. CBIR aims at avoiding the use of textual descriptions and instead retrieves images based on similarities in their contents (textures, colors, shapes etc.) to a user-supplied query image or user-specified image features.

List of CBIR Engines - list of engines which search for images-based image visual content such as color, texture, shape/object, etc.

Further information: Visual search engine and Reverse image search

Image collection exploration - search of images based on the use of novel exploration paradigms.

Survey based on Support Vector Machine (SVM):



Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Samples on the margin are called the support vectors.

A support-vector machine constructs a hyperplane or set of hyperplanes in a high or infinite dimensional space, which can be used for classification, regression, or other tasks like outliers detection. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin, the lower the generalized error of the classifier. Whereas the original problem may be stated in a finite-dimensional space, it often happens that the sets to discriminate are not linearly separable in that space. For this reason, it was proposed that the original finite-dimensional space be mapped into a much higher-dimensional space, presumably making the separation easier in that space. To keep the computational load reasonable, the mappings used by SVM schemes are designed to ensure that dot products of pairs of input data vectors may be computed easily in terms of the variables in the original space, by defining them in terms of a kernel function **k(x,y)** selected to suit the problem. The hyperplanes in the higher-dimensional space are defined as the set of points whose dot product with a vector in that space is constant, where such a set of vectors is an orthogonal (and thus minimal) set of vectors that defines a hyperplane. The vectors defining the hyperplanes can be chosen to be linear combinations with parameters $\boldsymbol{\alpha}_i$ of images of feature vector $\mathbf{x}_i$ that occur in the data base. With this choice of a hyperplane, the points **x** in the feature space that are mapped into the hyperplane are defined by the relation, $\sum_i \alpha_i \, k(x_i,x)=\text{constant}$ .Note that if **k(x,y)** becomes small as **y** grows further away from **x**, each term in the sum measures the degree of closeness of the test point **x** to the corresponding data base point $\mathbf{x}_i$. In this way, the sum of kernels above can be used to measure the relative nearness of each test point to the data points originating in one or the other of the sets to be discriminated. Note the fact that the set of points **x** mapped into any hyperplane can be quite convoluted as a result, allowing much more complex discrimination between sets that are not convex at all in the original space.

Convolutional Neural Networks (CNN):

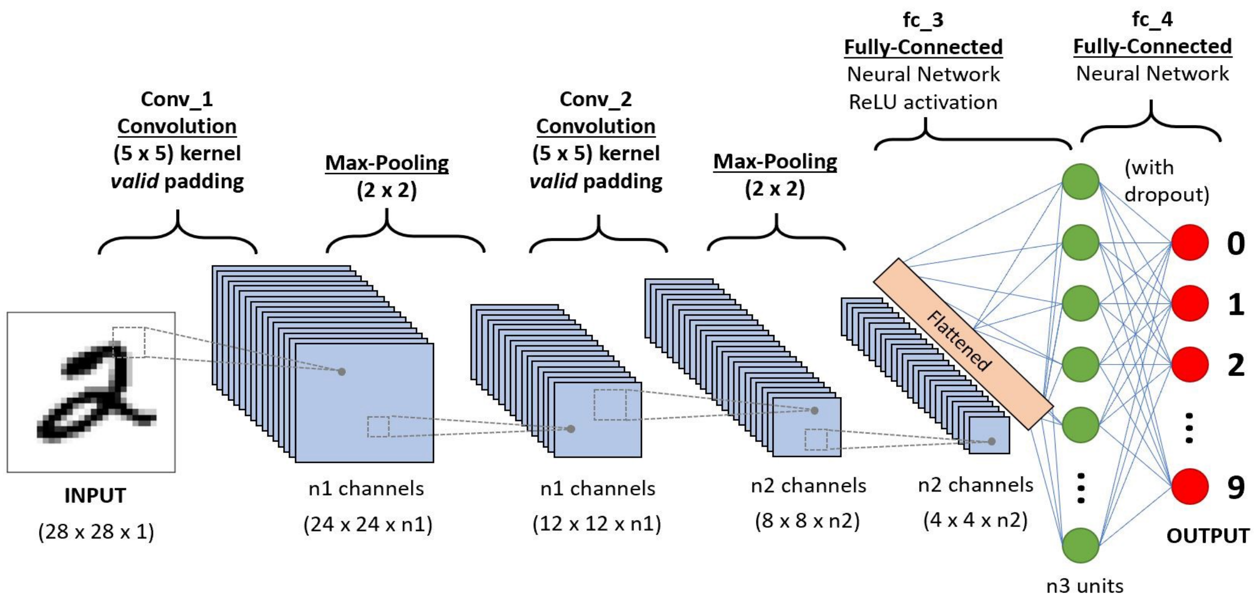


Fig: Architecture of CNN

A **Convolutional Neural Network** is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.NIST data set is having a huge number of handwritten text data set and it is frequently used for training, testing, and validation of CNN deep model. Researchers have created an efficient model with multiple convolutions, and pooling layers.

Chapter 4

METHODOLOGY

There are mainly four operation needed to build a simple ConvNet: Convolution, Non linearity, Pooling or Sub-sampling and classification (fully connected layer). These operations are the basic building block of every convolutional neural network channel is a term which used to refer a certain component of an image. A standard image will have three channels Red, Green, Blue. Each having pixel value in the range of 0 to 255. A gray scale image on the other hand has just one channel. These methods are explained below:

***The convolution step*:** Convolution neural network derive their name from the operator "convolution". The primary purpose of this operator in case of CNNs is to extract features from the input image. This is the first layer in CNN, the input to this layer is a 3D array (32*32*3) of pixel value. Convolution is a mathematical operation to merge two set of information in other our case, first is input image and the second set is the filter.

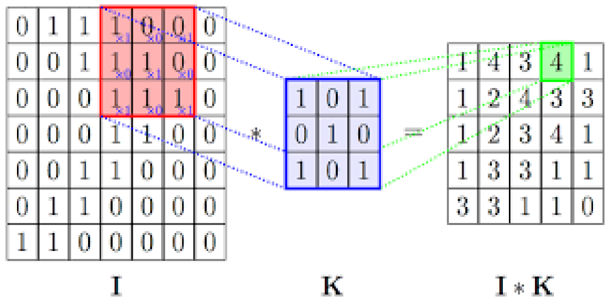


Fig: Convolution process

In figure, on the left side we have input image, convolution filter is located in the centre which is also called kernel, detector or feature whereas on the right side of figure 2 we have the output of the convolution process called activation map also called feature map, convolved feature. We perform convolution operation by sliding this filter over the input image. At every location we do an element wise matrix multiplication and sum the result, this resultant matrix is called Convolved feature. CNN learn the value of these filters on its own during training process.

Non linearity (ReLu): This is the addition operation called ReLu has been used after every convolution operation. ReLu is called rectified linear unit and is a non linear operation. It is given by: Output = Max(zero, input) as indicated in figure. ReLu is an element wise operation and replaces all negative pixel values in the feature map by zero.

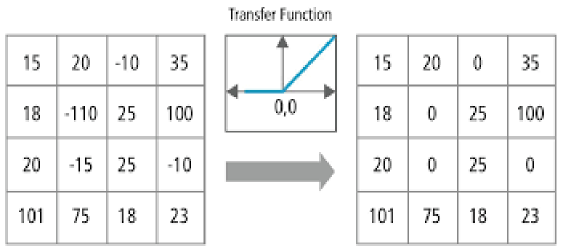


Fig: ReLu Operation

Pooling Step: Pooling (also called sub-sampling or down-sampling) it reduces the dimensionality of each feature map but retain the most important information. This layer makes the input representation smaller and more manageable. It also reduces the number of parameters and computations in the network and controlling from over fitting. The output of pooling layer acts as an input to the fully connected layer which is the next layer in the CNN after this. Figure shows the pooling operation in CNN architecture.

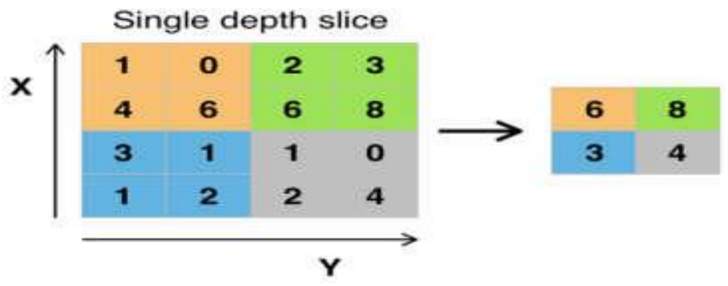


Fig: Pooling process

Fully connected layer: The fully connected layer is a traditional multilayer perceptron, it uses softmax activation function in the output layer. This term implies that every neuron in the previous layer is connected to every neuron in the next layer. The output from convolution layer and pooling layer represent high level features of image. The function of fully connected layer is to classifying these features of input image into various classes based on the training dataset. The sum of output probabilities from fully connected layer is 1. This is ensured by using softmax as the activation function in the output layer of fully connected layer.

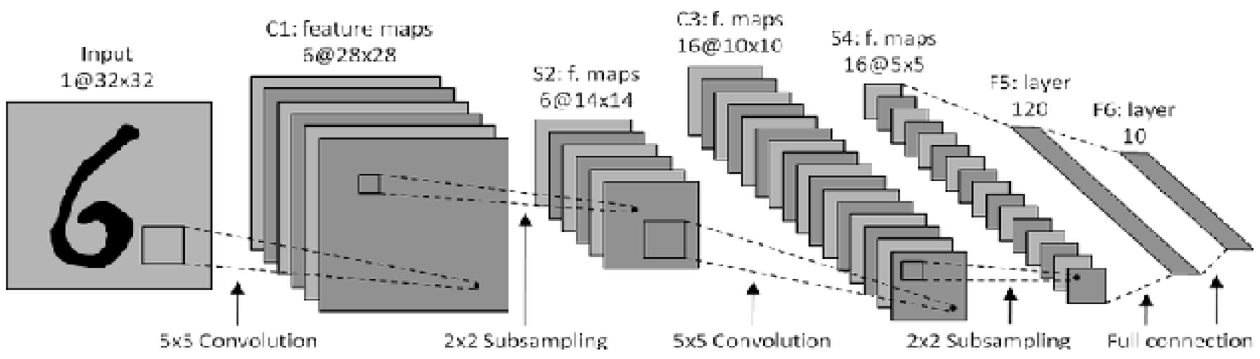


Fig: Fully connected layer

Following steps are involved in pre-processing the images before feeding it to the above trained CNN model:

1) **Pre-processing**:

This is the first step performed in image processing. In this step the noise from the image is removed by using median filtering. Median filtering is one of the most widely used noise reduction technique. This is because in median filtering the edges in image are preserved while the noise is still removed.

2) **Conversion to Gray-Scale**:

After the pre-processing step, the image is converted into grayscale. Conversion into grayscale is necessary because different writers use pens of different colors with varying intensities. Also working on grayscale images reduces the overall complexity of the system.

3) **Thresholding**:

When an image is converted into grayscale , the handwritten text is darker as compared to its background. With the help of thresholding we can seperate the darker regions of the image from the lighter regions. Thus because of thresholding we can seperate the handwritten text from its background.

4) **Image Segmentation**:

A user can write text in the form of lines. Thus the thresholded image is first segmented into individual lines .Then each individual line is segmented into individual words. Finally each word is segmented into individual characters. Segmentation of image into lines is carried out using Horizontal projection method. First the thresholded image is inverted so that background becomes

foreground and vice-versa. Now the image is scanned from top to bottom. While scanning, the sum of pixels in each row of image is calculated. The sum of pixels will be zero if all the pixels in one particular row are black. The sum will be non-zero if some white pixels are present in a row. After this a horizontal histogram is plotted in which the X-axis represents the
Y-coordinate of image (Starting from Top to Bottom) and the Y-axis represents the sum of pixels in the row corresponding to the Y-coordinate. The horizontal histogram can plot using MatPlotLib. The points marked in red in histogram are the points corresponding to the rows where sum of pixels are zero. After identifying all such rows, we can easily segment handwritten text into lines at these points. Now once the image is segmented into lines, each line must be further segmented into individual words. Segmentation of a line into words can be performed using the Vertical projection method. For segmenting line into words, we can make use of the fact that the spacing between two words is larger than the spacing between two characters. To segment a single line into individual words, the image is scanned from left to right and sum of pixels in each column is calculated. A vertical histogram is plotted in which the X-axis represents the X-coordinates of image and Y-axis represents the sum of pixels in each column. The points which are marked as red in the histogram are the points corresponding to the columns where sum of pixels is zero. The region where the sum of pixels is zero is wider when it is a region separating two words as compared to the region which is separating two characters. After segmenting a line into words, each word can be separated into individual character using similar technique as explained earlier. Now these individual characters are given to the pre-trained neural network model and predictions are obtained. Using this the final predicted text is sent back as a response to the user.
After the above processes the processed image is fed into the trained CNN model which takes the optical image as an input and recognize them and gives the output of the recognized texts.

Activation layers:

Activation functions are mathematical equations that determine the output of a neural network.

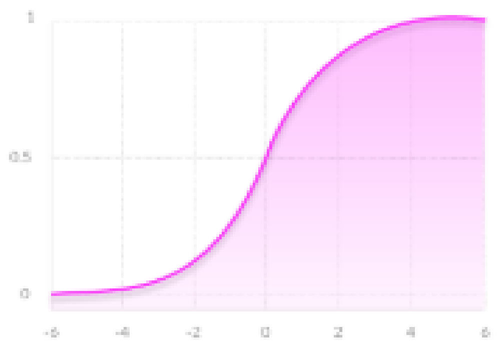
Activation functions also help normalize the output of each neuron to a range between 1 and 0 or between -1 and 1.

TYPES OF ACTIVATION FUNCTIONS REQUIRED:

THERE ARE THREE DIFFERENT ACTIVATION FUNCTIONS:

- SIGMOD ACTIVATION FUNCTION
- RELU ACTIVATION FUNCTION
- TANH ACTIVATION FUNCTION

SIGMOD ACTIVATION FUNCTION



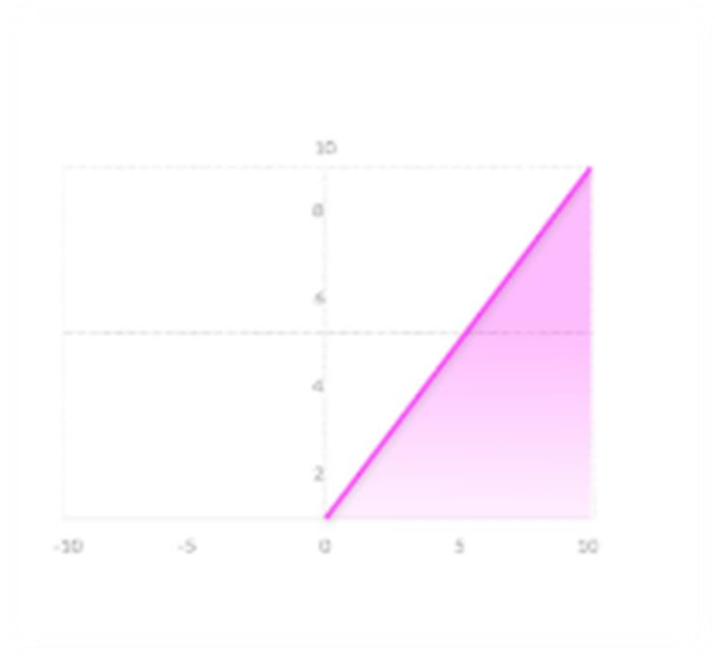
Advantages:

- Smooth gradient, preventing “jumps” in output values.
- Output values bound between 0 and 1, normalizing the output of each neuron.

Disadvantages:

- Vanishing gradient—for very high or very low values of X, there is almost no change to the prediction, causing a vanishing gradient problem. This can result in the network refusing to learn further or being too slow to reach an accurate prediction.
- Outputs not zero centered.
- Computationally expensive

RELU ACTIVATION FUNCTION:



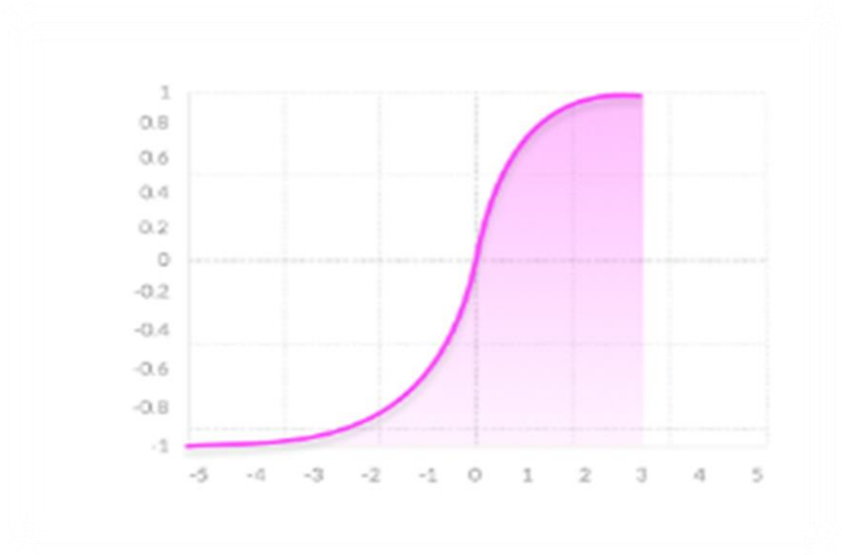
Advantages:

- Computationally efficient—allows the network to converge very quickly
- Non-linear—although it looks like a linear function, RELU has a derivative function and allows for backpropagation

Disadvantages:

- The Dying RELU problem—when inputs approach zero, or are negative, the gradient of the function becomes zero, the network cannot perform backpropagation and cannot learn.

TANH ACTIVATION FUNCTIONS:



Advantages:

- Zero centered—making it easier to model inputs that have strongly negative, neutral, and strongly positive values.
- Otherwise like the Sigmoid function.

Disadvantages:

- Same as the Sigmoid function

Chapter 5

PROJECT WORK DETAILS

History of Python

Python was conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to the ABC language (itself inspired by SETL), capable of exception handling and interfacing with the Amoeba operating system. Its implementation began in December 1989.Van Rossum shouldered sole responsibility for the project, as the lead developer, until July 12, 2018, when he announced his "permanent vacation" from his responsibilities as Python's *Benevolent Dictator For Life*, a title the Python community bestowed upon him to reflect his long-term commitment as the project's chief decision-maker. He now shares his leadership as a member of a five-person steering council. In January, 2019, active Python core developers elected Brett Cannon, Nick Coghlan, Barry Warsaw, Carol Willing and Van Rossum to a five-member "Steering Council" to lead the project.

Python 2.0 was released on 16 October 2000 with many major new features, including a cycle-detecting garbage collector and support for Unicode.

Python 3.0 was released on 3 December 2008. It was a major revision of the language that is not completely backward-compatible. Many of its major features were backported to Python 2.6.x and 2.7.x version series. Releases of Python 3 include the 2 to 3 utility, which automates (at least partially) the translation of Python 2 code to Python 3.

Python 2.7's end-of-life date was initially set at 2015 then postponed to 2020 out of concern that a large body of existing code could not easily be forward-ported to Python 3.

Over view of python

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

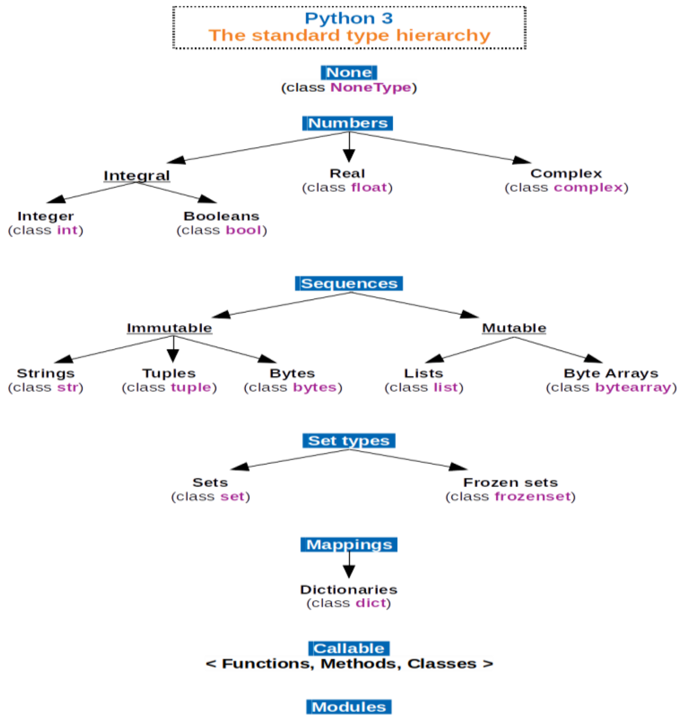


Fig: Overview of Hierarchy in python

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Features and philosophy

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by metaprogramming and metaobjects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution.

Python's design offers some support for functional programming in the Lisp tradition. It has filter, map, and reduce functions; list comprehensions, dictionaries, sets, and generator expressions. The standard library has two modules (itertools and functools) that implement functional tools borrowed from Haskell and Standard ML.

The language's core philosophy is summarized in the document *The Zen of Python* (*PEP 20*), which includes aphorisms such as:

1) Beautiful is better than ugly.
2) Explicit is better than implicit.
3) Simple is better than complex.
4) Complex is better than complicated.
5) Readability counts.

Rather than having all of its functionality built into its core, Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach.

Python strives for a simpler, less-cluttered syntax and grammar while giving developers a choice in their coding methodology. In contrast to Perl's "there is more than one way to do it" motto, Python embraces a "there should be one—and preferably only one—obvious way to do it" design philosophy. Alex Martelli, a Fellow at the Python Software Foundation and Python book author, writes that "To describe something as 'clever' is *not* considered a compliment in the Python culture."

Python's developers strive to avoid premature optimization, and reject patches to non-critical parts of the CPython reference implementation that would offer marginal increases in speed at the cost of clarity. When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C, or use PyPy, a just-in-time compiler. CPython is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter.

An important goal of Python's developers is keeping it fun to use. This is reflected in the language's name—a tribute to the British comedy group Monty Python—and in occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (from a famous Monty Python sketch) instead of the standard foo and bar.

A common neologism in the Python community is *pythonic*, which can have a wide range of meanings related to program style. To say that code is pythonic is to say that it uses Python idioms well, that it is natural or shows fluency in the language, that it conforms with Python's minimalist philosophy and emphasis on readability. In contrast, code that is difficult to understand or reads like a rough transcription from another programming language is called *unpythonic*.

Users and admirers of Python, especially those considered knowledgeable or experienced, are often referred to as *Pythonistas*.

Python modules Overview

Modular programming refers to the process of breaking a large, unwieldy programming task into separate, smaller, more manageable subtasks or **modules**. Individual modules can then be cobbled together like building blocks to create a larger application.

There are several advantages to **modularizing** code in a large application:

Simplicity: Rather than focusing on the entire problem at hand, a module typically focuses on one relatively small portion of the problem. If you're working on a single module, you'll have a smaller problem domain to wrap your head around. This makes development easier and less error-prone.

Maintainability: Modules are typically designed so that they enforce logical boundaries between different problem domains. If modules are written in a way that minimizes interdependency, there is decreased likelihood that modifications to a single module will have an impact on other parts of the program. (You may even be able to make changes to a module without having any knowledge of the application outside that module.) This makes it more viable for a team of many programmers to work collaboratively on a large application.

Reusability: Functionality defined in a single module can be easily reused (through an appropriately defined interface) by other parts of the application. This eliminates the need to recreate duplicate code.

Scoping: Modules typically define a separate **namespace**, which helps avoid collisions between identifiers in different areas of a program. (One of the tenets in the Zen of Python is *Namespaces are one honking great idea—let's do more of those!*)

Functions, **modules** and **packages** are all constructs in Python that promote code modularization.

Important Python modules

OpenCV:

OpenCV (Open source computer vision) 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 BSD license. OpenCV supports the deep learning frameworks such as TensorFlow.

OpenCV can be used to perform various image processing operations like segmentation, thresholding and Morphological Operations. It's a open source library used for Image processing.

NumPy:

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

→a powerful N-dimensional array object

→sophisticated (broadcasting) functions

→tools for integrating C/C++ and Fortran code

→useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases. We need numpy to work with the data arrays.

Tensorflow:

TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications. Tensorflow will be the key component, as we are using it to create the AI.

Math:

The math module is a standard module in Python and is always available. To use mathematical functions under this module, you have to import the module using import math.It gives access to the underlying C library functions. Math provides some neat helper functions.

Matplotlib:

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy.It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+.

Other packages:

Some of other packages used in the python which will be helpful to build models are sys, time ,datetime ,os. We need sys to read in the arguments. We will use that to specify the amount of training cycles via the command line. time and datetime are used for logging and os is used to set the tensorflow debug level.

CONCLUSION

An overview of convolutional neural network and working of all layers is presented in accordance with other techniques and models. And the above survey of different models concludes that the different parameters like dropout, optimizer, learning rates greatly influence the efficiency of the CNN architectures. Preprocessing of dataset also consider a major factor behind the accuracy of the models. Training of network requires a supportive hardware to implement large dataset efficiently. In some language the characters are differ by only a single dot for that characters it is necessary to train the network with large amount of dataset so that the network easily recognize the character for that there is a requirement of high memory and high processing speed to achieve a efficient network. The articles provided in the literature survey contributes different networks with different tuning of parameters on different types of dataset which help in achieving efficient network in future that can even recognize a whole sentence of different handwritten languages.

REFERENCES

- Rui Pedro da Silva Rodrigues Machado, Dissertation for achieving the degree of master’s in data Analytics IEEE, 2017.

- Clothing image retrieval based on a similarity evaluation method for Kansei retrieval system by Takaki Urai and Daichi Okunaka Department of Science and Engineering Graduate School of Kansai University Osaka, Japan, IEEE, 2012
- . http://en.wikipedia.org/wiki/Recommender_system
- Neural Networks and Deep Learning by Michael Nielsen
- TensorFlow Deep Learning Cookbook by Gulli and Kapoor's
- Deep Learning For Computer Vision With Python by Francois Chollet-Creator of Keras
-