**A Seminar Report**

On

PATTERN RECOGNITION

(Using artificial intelligence)

**BACHELOR OF TECHNOLOGY**

*In*

COMPUTER SCIENCE ENGINEERING

*By*

ANURAG & AMISHA KALRA

CSE/16/108 & CSE/16/107



Submitted to

Department of COMPUTER SCIENCE ENGINEERING

B.M Institute of Engineering and Technology

SONEPAT, Haryana (INDIA)

September,2018

**ACKNOWLEDEGEMENT**

It would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I am highly indebted to (**HPE**) for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

I would like to express my gratitude towards **KYROIN** and **HPE** for their kind co-operation and training which help me in completion of this project.

I would like to express my special gratitude and thanks to industry persons for giving me such attention and time.

My thanks and appreciations also go to my colleague in developing the project and people who have willingly helped me out with their abilities.





****

**CERTIFICATE**

Anurag & Amisha kalra are Students of B.Tech (CSE), 3rd year of **B.M Institute of Engineering and technology, Sonepat** (Haryana) completed my summer In-house training from **KYROIN associated with HPE CoC certification** from June 18, 2018 to June 27, 2018.

During the mentioned period I worked at artificial intelligence using python and completed my summer training project entitled “**PATTERN RECOGINITON**” under the guidance of my teacher.

Date: Name: Anurag & Amisha Kalra

**B.M Institute of Engineering and Technology**

**Sonepat**

**STUDENT’S DECLARATION**

Anurag & Amisha Kalra hereby declare that I have undertaken two weeks industrial training at **KYROIN** with **HPE** during a period from June 18,2018 to June 27,2018 in partial fulfilment of requirements for the award of degree of B.Tech (Computer Science Engineering) at **B.M Institute of Engineering and Technology**,**Sonepat**. The work which is being presented in the training report submitted to Department of **Computer Science Engineering**at **B.M Institute of Engineering and Technology, Sonepat** is an authentic record of training work.

Signature of student

Anurag & Amisha Kalra

CSE/16/108

The two weeks industrial training **Viva-Voice Examination** of **python** has been held on

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ and accept

Signature of Internal Examiner Signature of External Examiner

**Table of Contents**

**Acknowledgement**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2

**Certificate**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4

**Student Declaration**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5

**Table of Contents**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_6

1. **Company Profile**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_8

2. **System Requirements**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_13

2.1. Hardware Interface\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_13

2.2. Software Interface\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_13

3. **Technology**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_14

3.1. **Software Used**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_14

3.2. **Introduction to Python**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_9

3.1.1. What is Python\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_9

3.1.2. What can Python do\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_9

3.1.3. Python Install\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_10

3.1.4. Variables Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_10

3.1.5. Types of Variables \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_11

3.3. **Introduction to artificialintelligence**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_12

3.2.1. What is AI \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_12

3.2.2. Where did it come from\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_13

3.2.3. How popular is AI\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_14

3.2.4. What are the applications \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_14

3.2.5 What else can do\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20

4. **Project**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_21

4.1. **Introduction to Project**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_21

4.1.1. What is pattern recognition \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_21

4.1.2. Why should affiliates to blog\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_21

4.1.3. Downsides to blogging\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_22

4.1.4. How do I Blog\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_22

4.2. **Source Code**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_23

4.3. **Outputs**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_63

5. **References**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_6

1.**COMPANY PROFILE**

1.1 KYRION

**Kyrion Digital Securities**, Flagship Company of Kyrion Group, is working towards the vision to develop India as a **Cyber Secured Country**. It is a leading player in India to provide Ethical Hacking & IT Security Training to all possible audience. With an outreach to over eighty thousand students and over a thousand major colleges, Kyrion Digital Securities is a well known and trusted brand in the education and information security business.

Kyrion Digital Securities has a wide and vibrant profile which caters to a range of professional courses that are customized to suit the aspirations of different graduate streams. The USP of the company lies in its ability to clean and sustain itself, it is supported by a clear and strong business model that focuses of improvising to adjust with the trends and needs of a volatile market.

**A BRIEF TABULAR DESCRIPTION OF KYRION DIGITAL SECURITIES IS STATED:**

* Digital Security is the need of an hour. Kyrion provides training and education in the field of Ethical Hacking & Information Security to the students from schools and colleges as well as to corporate executives. Trainings can be availed at client location (Onsite Training), at Kyrion Training Center (Offsite Training) or through distance mode.
* The IT sector is currently facing a shortage in Security Solutions. Kyrion's home-grown concept of IT with Security Plugins has proved to be an exemplary solution. Kyrion aims to create a better workforce which could support and shape the evolving sector of IT security. In this way, Kyrion helps the IT sector by providing efficient and cost-effective manpower for their businesses.
* Engineering & MCA colleges are aplenty, however a flip side of this truth is that there is a dearth of trained and motivated faculty, because of which students are unable to meet the corporate requirements of the market. Kyrion with its initiative Campus2Corporate aims to bridge the gap between theoretical knowledge acquired from university and the ability to tackle the day-to-day problems faced by the industry.

**THEIR HEAD OFFICES ARE LOCATED AT ALL MAJOR LOCATIONS SUCH AS:**

* Connaught Place Delhi, India
* Pennsylvania, United State of America (USA)
* London, United Kingdom (UK)
* Perth, Western Australia

Team at Kyrion believes that opportunities are borne out of imperfections. Our **Education Entrepreneurship Firm**, aims to deliver high-end security solutions and training to students and corporates to prepare them to meet the challenges of IT Security Industry. Kyrion also set the benchmark in the industry by being the premier entity that introduced the concept of IT with Security Plugins in India and since then, the Indian IT sector has soared higher.

One of our major challenges was to sensitize the professionals and students about the need to master IT Security skill set. Kyrion aimed at building a sustainable culture of trained security professionals that would help the industries reduce their dependency upon foreign experts and solutions. An extensive network of Indian Defense (Navy and Air Force), 1000 engineering colleges, 102 schools and 62 corporate, government & defense organizations in this short span of time makes us feel proud of our endeavors and motivate us to continue on the chosen path. We are one of the fastest growing and globally recognized IT Securities firm for conducting workshops and corporate training in India and world-wide.

# 1.2VISION

Our efforts are focused on bringing out the best within every individual and epitomizing his capacity to unprecedented levels.

Almost all the developed nations surpass our country in the genre of technology.This is the reason why the technical comfort of Indians remain at an all time low. The primary reason is the skepticism to try out something new. But the trend is laterally changing and this gives us hope.

Our aim is to create a foolproof and impeccably secure information system in India to guard against all kind of malicious information security breach. Thus, empowering the nation with reliable, efficient and secure database systems.

We have planned to reach those heights through meticulous frameworks and action plans. Our steps begin at university levels and reach out the India Inc and Highly Technical Government Services.

Kyrion Digital Securities Pvt. Ltd is building a work force that experts in building secure and safe data gateways and storage. We refine your information models to emancipate any chance of malicious penetration and render your critical data safe and secure.

# 1.3 MISSION AND OBJECTIVE

Our mission is to render every database based transaction safe, secure and simple. We aim to transform the internet security industry by infusing professionalism and a never before seen efficiency.

With many years of experience and meticulous expertise we are here to safeguard your interests and critical data from online and offline threats or vulnerabilities.

We provide cutting edge internet security solutions to help our customers achieve unprecedented levels of smoothness and productivity.

Alongside we dream to create a parallel equipped workforce that will shape the future in the business of internet securities. Our aim is to make every technical student an ace in internet securities.

Kyrion Digital Securities Pvt. Ltd.. Is dedicated to provide you efficient, cost effective and dependable security solutions that will forever change the way you do your business.

1.4COURSES

Kyrion technologies has a wide and vibrant profile which caters to a range of professional courses that are customized to suit the aspirations of different graduate streams.

ETHICAL HACKING

APP DEVELOPMENT

EMBEDDED SYSTEM

SOFTWARE DEVELOPMENT

PRODUCT DESIGNING

BIG DATA

# 1.5CONTACTS

## Kyrion Technologies Pvt. Ltd.

**Delhi, India**

**CP:** 1009, Indraprakash Building, Barakhamba Road, Connaught Place, New Delhi - 110001   
**PITAMPURA:** First Floor, 21 Kapil Vihar, Pitampura, Opp Metro Pillar 352, New Delhi 110034  
Tel: +91-95401-12244/+91-95401-19933

**UNITED STATE OF AMERICA (USA)**  
Kyrion US  
22204, Cornerstone Drive, Yardley, Pennsylvania 19067, USA  
Tel: +1-215-300-1235

**UNITED KINGDOM (UK)**  
Kyrion UK  
317 Nether Street, Finchley Central, N3 1JJ, London, UK  
Tel: +44-75-4006-3332

**AUSTRALIA**  
Kyrion Australia  
14/413 Canning Highway, Como, Perth, Western Australia 6152  
Tel: +61-44-886-7464

## Regd. Office

**Kyrion Technologies Pvt. Ltd.**B-92, G.T Karnal Road, Industrial Area, Delhi-110033 (India)   
**Contact No.:** +91-98181-98188   
**E-mail:**[info@kyrion.in](mailto:info@kyrion.in)

**A unit of Kyrion Technologies Pvt. Ltd.  
Tel: 011-4355-9494  
Email: info@kyrion.**

#### 1.6ASSOCIATION WITH : HPE

#### **Hewlett Packard (HP) Certified** **Summer Training**

### **ABOUT HP COC SUMMER TRAINING PROGRAM**

**The HP CoC Summer Training program** focuses on covering **basic to**

**intermediate level** of respective technologies that are part of the training program. The courses are designed in accordance to **practical and latest industrial practices** taught by experts from the field. **Project based** training would further enhance the learning process.

**2.SYSTEM REQUIREMENTS**

2.1. Hardware Interface:

* Pentium(R) 4 CPU 2.26GHz, 128MB RAM
* Screen Resolution of at least 800 x 600 required for proper and complete viewing of screens. Higher Resolution would not be a problem.

2.2. Software Interface:

* Any Window based operating System (Windows 95/98/2000/XP/NT/7/8/10)

**3. INTRODUCTION TO TECHNOLOGY**

3.1. **SOFTWARE USED**

Software in use: Python 2.7

Python IDLE: **IDLE** (short for integrated development environment or integrated development and learning environment) is an integrated development environment for **Python**, which has been bundled with the default implementation of the language since 1.5.2b1.

Downloaded fromwww.pyhton.org

Documentation at: www.docs.python.org

3.2. **INTRODUCTION TO PYTHON**

3.2.1. WHAT IS PYTHON?

Python is a popular programming language. It was created in 1991 by Guido van Rossum.

It is used for:

* web development (server-side),
* software development,
* mathematics,
* system scripting.

3.2.2. WHAT CAN PYTHON DO?

* Python can be used on a server to create web applications.
* Python can be used alongside software to create workflows.
* Python can connect to database systems. It can also read and modify files.
* Python can be used to handle big data and perform complex mathematics.
* Python can be used for rapid prototyping, or for production-ready software development.

3.2.3. PYTHON INSTALL

Many PCs and Macs will have python already installed.

To check if you have python installed on a Windows PC, search in the start bar for Python or run the following on the Command Line (cmd.exe):

**C:\Users\Pcname>python –version**

To check if you have python installed on a Linux or Mac, then on linux open the command line or on Mac open the Terminal and type:

**python –version**

3.2.4PYTHON QUICKSTART

Python is an interpreted programming language, this means that as a developer you write Python (.py) files in a text editor and then put those files into the python interpreter to be executed.

The way to run a python file is like this on the command line:

**C:\Users\*Your Name*>python helloworld.py**

Where "helloworld.py" is the name of your python file.

Let's write our first Python file, called helloworld.py, which can be done in any text editor.

**helloworld.py**

**print("Hello, World!")**

Simple as that. Save your file. Open your command line, navigate to the directory where you saved your file, and run:

**C:\Users\*Your Name*>python helloworld.py**

The output should read:

**Hello, World!**

3.2.5. VARIABLES NAME

A variable can have a short name (like x and y) or a more descriptive name (age, carname, total\_volume). Rules for Python variables:

* A variable name must start with a letter or the underscore character
* A variable name cannot start with a number
* A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and \_ )
* Variable names are case-sensitive (age, Age and AGE are three different variables)

**Remember that variables are case-sensitive**

## 3.2.6. SPECIFY A VARIABLE TYPE

There may be times when you want to specify a type on to a variable. This can be done with casting. Python is an object-orientated language, and as such it uses classes to define data types, including its primitive types.

Casting in python is therefore done using constructor functions:

* int() - constructs an integer number from an integer literal, a float literal (by rounding down to the previous whole number), or a string literal (providing the string represents a whole number)
* float() - constructs a float number from an integer literal, a float literal or a string literal (providing the string represents a float or an integer)
* str() - constructs a string from a wide variety of data types, including strings, integer literals and float literals

### **EXAMPLE**

Integers:

x = int(1)   # x will be 1

Floats:

x = float(1)     # x will be 1.0

Strings:

x = str("s1") # x will be 's1'

3.3. **INTRODUCTION TO ARTIFICIAL INTELLIGENCE**

3.3.1. **What is ARTIFICIAL INTELLIGENCE?**

3.3.3. **WHAT WE CAN DO USING AI?**

**Artificial intelligence (AI), sometimes called machine intelligence, is intelligence demonstrated by machines, in contrast to the natural intelligence displayed by humans and other animals. In computer science AI research is defined as the study of "intelligent agents": any device that perceives its environment and takes actions that maximize its chance of successfully achieving its goals. Colloquially, the term "artificial intelligence" is applied when a machine mimics "cognitive" functions that humans associate with other human minds, such as "learning" and "problem solving".**

**The scope of AI is disputed: as machines become increasingly capable, tasks considered as requiring "intelligence" are often removed from the definition, a phenomenon known as the AI effect, leading to the quip, "AI is whatever hasn't been done yet." For instance, optical character recognition is frequently excluded from "artificial intelligence", having become a routine technology. Modern machine capabilities generally classified as AI include successfully understanding human speech, competing at the highest level in strategic game systems (such as chess and Go), autonomously operating cars, and intelligent routing in content delivery networks and military simulations.**

**Artificial intelligence was founded as an academic discipline in 1956, and in the years since has experienced several waves of optimism, followed by disappointment and the loss of funding (known as an "AI winter"), followed by new approaches, success and renewed funding. For most of its history, AI research has been divided into subfields that often fail to communicate with each other. These sub-fields are based on technical considerations, such as particular goals (e.g. "robotics" or "machine learning"), the use of particular tools ("logic" or artificial neural networks), or deep philosophical differences. Subfields have also been based on social factors (particular institutions or the work of particular researchers).**

**3.3.2. HISTORY.**

**Thought-capable artificial beings appeared as storytelling devices in antiquity, and have been common in fiction, as in Mary Shelley's Frankenstein or Karel Čapek'sR.U.R. (Rossum's Universal Robots). These characters and their fates raised many of the same issues now discussed in the ethics of artificial intelligence.**

**The study of mechanical or "formal" reasoning began with philosophers and mathematicians in antiquity. The study of mathematical logic led directly to Alan Turing's theory of computation, which suggested that a machine, by shuffling symbols as simple as "0" and "1", could simulate any conceivable act of mathematical deduction. This insight, that digital computers can simulate any process of formal reasoning, is known as the Church–Turing thesis. Along with concurrent discoveries in neurobiology, information theory and cybernetics, this led researchers to consider the possibility of building an electronic brain. Turing proposed that "if a human could not distinguish between responses from a machine and a human, the machine could be considered “intelligent". The first work that is now generally recognized as AI was McCullouch and Pitts' 1943 formal design for Turing-complete"artificial neurons".**

**The field of AI research was born at a workshop at Dartmouth College in 1956.[28] Attendees Allen Newell (CMU), Herbert Simon (CMU), John McCarthy (MIT), Marvin Minsky (MIT) and Arthur Samuel (IBM) became the founders and leaders of AI research. They and their students produced programs that the press described as "astonishing": computers were learning checkers strategies (c. 1954) (and by 1959 were reportedly playing better than the average human), solving word problems in algebra, proving logical theorems (Logic Theorist, first run c. 1956) and speaking English. By the middle of the 1960s, research in the U.S. was heavily funded by the Department of Defense and laboratories had been established around the world. AI's founders were optimistic about the future: Herbert Simon predicted, "machines will be capable, within twenty years, of doing any work a man can do". Marvin Minsky agreed, writing, "within a generation ... the problem of creating 'artificial intelligence' will substantially be solved".**

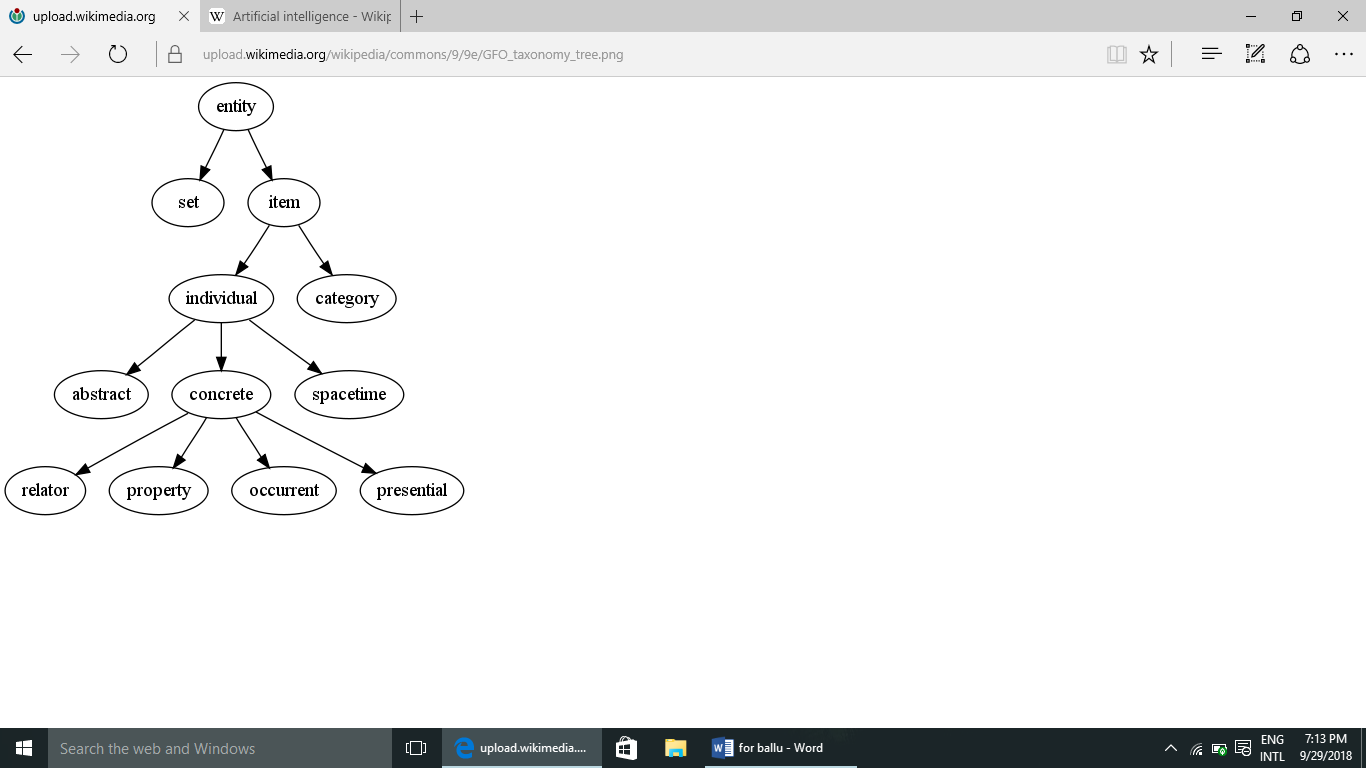
**They failed to recognize the difficulty of some of the remaining tasks. Progress slowed and in 1974, in response to the criticism of Sir James Lighthilland ongoing pressure from the US Congress to fund more productive projects, both the U.S. and British governments cut off exploratory research in AI. The next few years would later be called an "AI winter", a period when obtaining funding for AI projects was difficult.**

**In the early 1980s, AI research was revived by the commercial success of expert systems, a form of AI program that simulated the knowledge and analytical skills of human experts. By 1985, the market for AI had reached over a billion dollars. At the same time, Japan's fifth generation computer project inspired the U.S and British governments to restore funding for academic research. However, beginning with the collapse of the Lisp Machine market in 1987, AI once again fell into disrepute, and a second, longer-lasting hiatus began.**

**In the late 1990s and early 21st century, AI began to be used for logistics, data mining, medical diagnosis and other areas.[22] The success was due to increasing computational power (see Moore's law), greater emphasis on solving specific problems, new ties between AI and other fields (such as statistics, economics and mathematics), and a commitment by researchers to mathematical methods and scientific standards.Deep Blue became the first computer chess-playing system to beat a reigning world chess champion, Garry Kasparov on 11 May 1997.**

**According to Bloomberg's Jack Clark, 2015 was a landmark year for artificial intelligence, with the number of software projects that use AI within Google increased from a "sporadic usage" in 2012 to more than 2,700 projects. Clark also presents factual data indicating that error rates in image processing tasks have fallen significantly since 2011. He attributes this to an increase in affordable neural networks, due to a rise in cloud computing infrastructure and to an increase in research tools and datasets. Other cited examples include Microsoft's development of a Skype system that can automatically translate from one language to another and Facebook's system that can describe images to blind people. In a 2017 survey, one in five companies reported they had "incorporated AI in some offerings or processes**

### **3.3.3.1 KNOWLEDGE REPRESENTATION**:

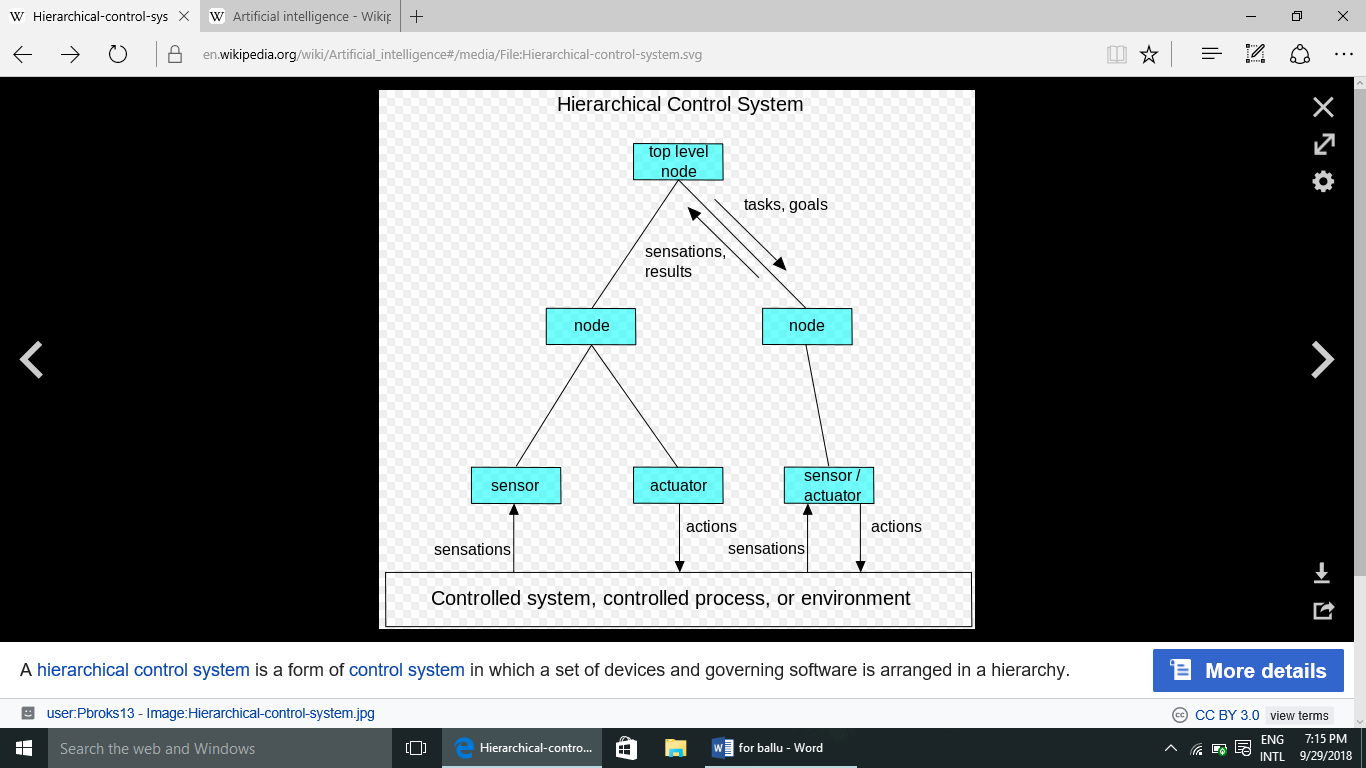


An ontology represents knowledge as a set of concepts within a domain and the relationships between those concepts.

*Main articles: Knowledge representation and Commonsense knowledge*

Knowledge representation and knowledge engineering are central to classical AI research. Some "expert systems" attempt to gather together explicit knowledge possessed by experts in some narrow domain. In addition, some projects attempt to gather the "commonsense knowledge" known to the average person into a database containing extensive knowledge about the world. Among the things a comprehensive commonsense knowledge base would contain are: objects, properties, categories and relations between objects; situations, events, states and time; causes and effects; knowledge about knowledge (what we know about what other people know); and many other, less well researched domains. A representation of "what exists" is an ontology: the set of objects, relations, concepts, and properties formally described so that software agents can interpret them. The semantics of these are captured as description logic concepts, roles, and individuals, and typically implemented as classes, properties, and individuals in the Web Ontology Language. The most general ontologies are called upper ontologies, which attempt to provide a foundation for all other knowledge by acting as mediators between domain ontologies that cover specific knowledge about a particular knowledge domain (field of interest or area of concern). Such formal knowledge representations can be used in content-based indexing and retrieval,scene interpretation, clinical decision support, knowledge discovery (mining "interesting" and actionable inferences from large databases), and other areas.

### **3.3.3.2 PLANNING**



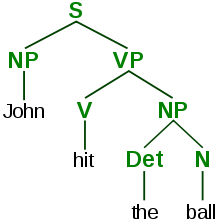
A hierarchical control system is a form of control system in which a set of devices and governing software is arranged in a hierarchy.

*Main article: Automated planning and scheduling*

Intelligent agents must be able to set goals and achieve them. They need a way to visualize the future—a representation of the state of the world and be able to make predictions about how their actions will change it—and be able to make choices that maximize the utility (or "value") of available choices.In classical planning problems, the agent can assume that it is the only system acting in the world, allowing the agent to be certain of the consequences of its actions. However, if the agent is not the only actor, then it requires that the agent can reason under uncertainty. This calls for an agent that can not only assess its environment and make predictions, but also evaluate its predictions and adapt based on its assessment.

Multi-agent planning uses the cooperation and competition of many agents to achieve a given goal. Emergent behavior such as this is used by evolutionary algorithms and swarm intelligence.

### **3.3.3.3 NATURAL LANGUAGE PROCESSING**



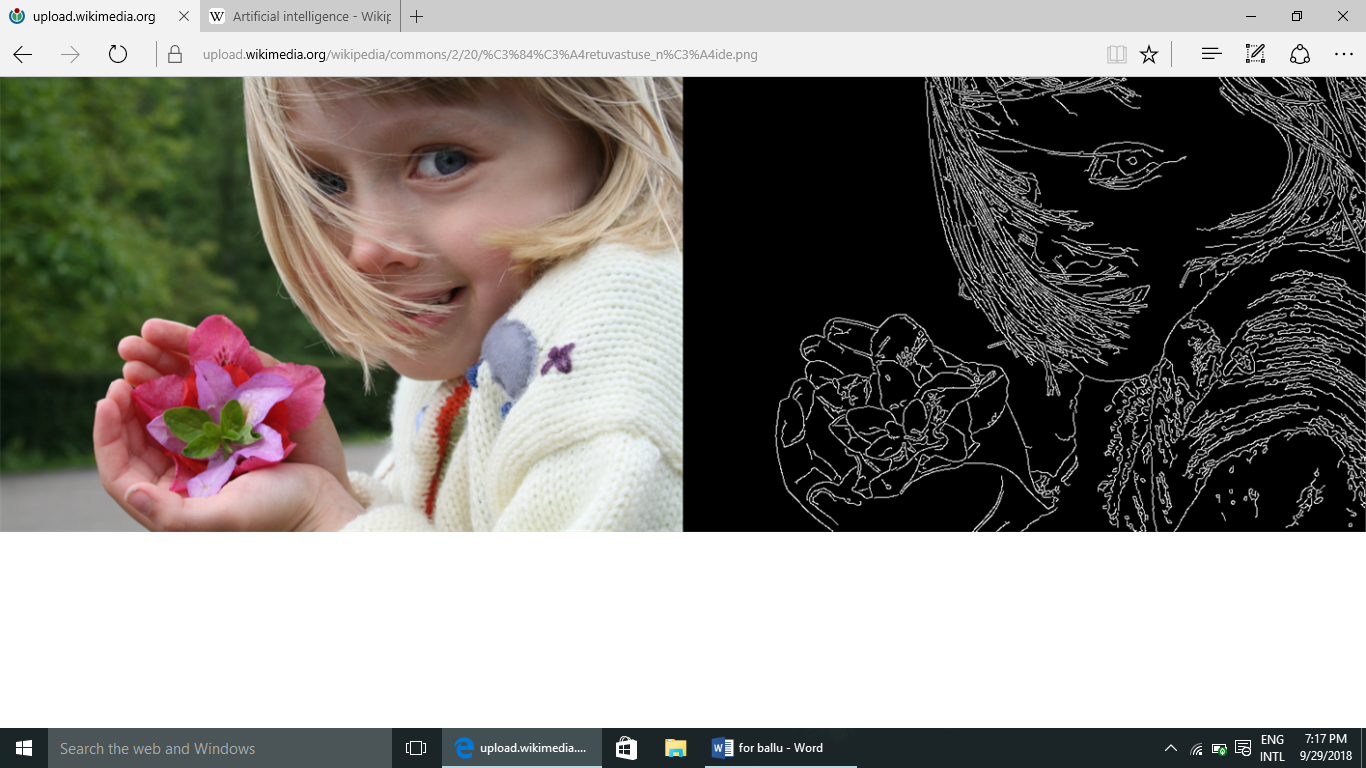
A parse tree represents the syntactic structure of a sentence according to some formal grammar.

*Main article: Natural language processing*

Natural language processing(NLP) gives machines the ability to read and understand human language. A sufficiently powerful natural language processing system would enable natural-language user interfaces and the acquisition of knowledge directly from human-written sources, such as newswire texts. Some straightforward applications of natural language processing include information retrieval, text mining, question answering and machine translation. Many current approaches use word co-occurrence frequencies to construct syntactic representations of text. "Keyword spotting" strategies for search are popular and scalable but dumb; a search query for "dog" might only match documents with the literal word "dog" and miss a document with the word "poodle". "Lexical affinity" strategies use the occurrence of words such as "accident" to assess the sentiment of a document. Modern statistical NLP approaches can combine all these strategies as well as others, and often achieve acceptable accuracy at the page or paragraph level, but continue to lack the semantic understanding required to classify isolated sentences well. Besides the usual difficulties with encoding semantic commonsense knowledge, existing semantic NLP sometimes scales too poorly to be viable in business applications. Beyond semantic NLP, the ultimate goal of "narrative" NLP is to embody a full understanding of commonsense reasoning.

### **3.3.3.4 PERCEPTION**

*Main articles: Machine perception, Computer vision, and Speech recognition*



Feature detection (pictured: edge detection) helps AI compose informative abstract structures out of raw data.

Machine perception is the ability to use input from sensors (such as cameras (visible spectrum or infrared), microphones, wireless signals, and active lidar, sonar, radar, and tactile sensors) to deduce aspects of the world. Applications include speech recognition, facial recognition, and object recognition. Computer vision is the ability to analyze visual input. Such input is usually ambiguous; a giant, fifty-meter-tall pedestrian far away may produce exactly the same pixels as a nearby normal-sized pedestrian, requiring the AI to judge the relative likelihood and reasonableness of different interpretations, for example by using its "object model" to assess that fifty-meter pedestrians do not exist.

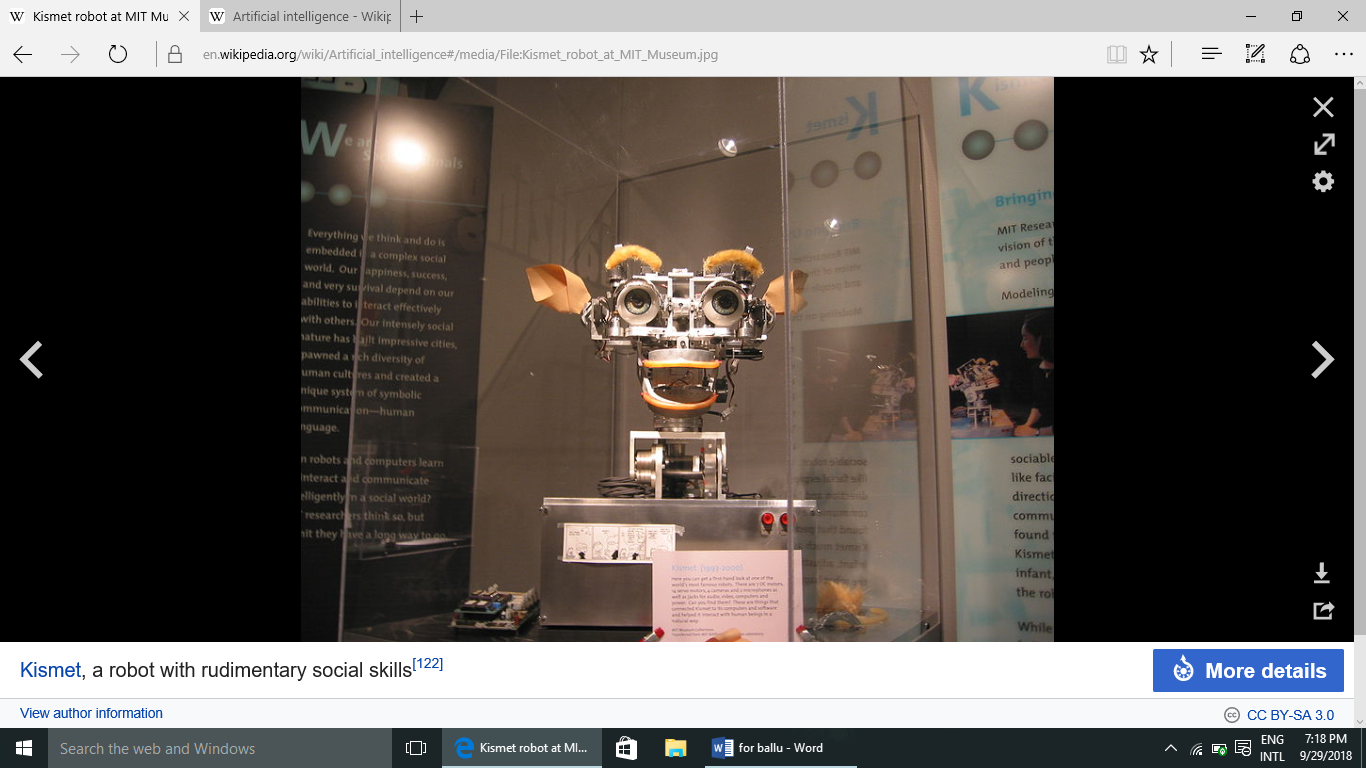
### **3.3.3.5 MOTION AND MANIPULATION**

*Main article: Robotics*

AI is heavily used in robotics. Advanced robotic arms and other industrial robots, widely used in modern factories, can learn from experience how to move efficiently despite the presence of friction and gear slippage. A modern mobile robot, when given a small, static, and visible environment, can easily determine its location and map its environment; however, dynamic environments, such as (in endoscopy) the interior of a patient's breathing body, pose a greater challenge. Motion planning is the process of breaking down a movement task into "primitives" such as individual joint movements. Such movement often involves compliant motion, a process where movement requires maintaining physical contact with an object. Moravec's paradox generalizes that low-level sensorimotor skills that humans take for granted are, counterintuitively, difficult to program into a robot; the paradox is named after Hans Moravec, who stated in 1988 that "it is comparatively easy to make computers exhibit adult level performance on intelligence tests or playing checkers, and difficult or impossible to give them the skills of a one-year-old when it comes to perception and mobility". This is attributed to the fact that, unlike checkers, physical dexterity has been a direct target of natural selection for millions of years.

### **3.3.3.6 SOCIAL INTELLIGENCE**

*Main article: Affective computing*



[Kismet](https://en.wikipedia.org/wiki/Kismet_(robot)), a robot with rudimentary social skills

Moravec's paradox can be extended to many forms of social intelligence.Distributed multi-agent coordination of autonomous vehicles remains a difficult problem.Affective computing is an interdisciplinary umbrella that comprises systems which recognize, interpret, process, or simulate human affects. Moderate successes related to affective computing include textual sentiment analysis and, more recently, multimodal affect analysis (see multimodal sentiment analysis), wherein AI classifies the affects displayed by a videotaped subject.

In the long run, social skills and an understanding of human emotion and game theory would be valuable to a social agent. Being able to predict the actions of others by understanding their motives and emotional states would allow an agent to make better decisions. Some computer systems mimic human emotion and expressions to appear more sensitive to the emotional dynamics of human interaction, or to otherwise facilitate human–computer interaction. Similarly, some virtual assistants are programmed to speak conversationally or even to banter humorously; this tends to give naïve users an unrealistic conception of how intelligent existing computer agents actually are.

3.3.4. WHAT ARE THE APPLICATIONS?

An automated online assistant providing customer service on a web page – one of many very primitive applications of artificial intelligence

Main article: Applications of artificial intelligence

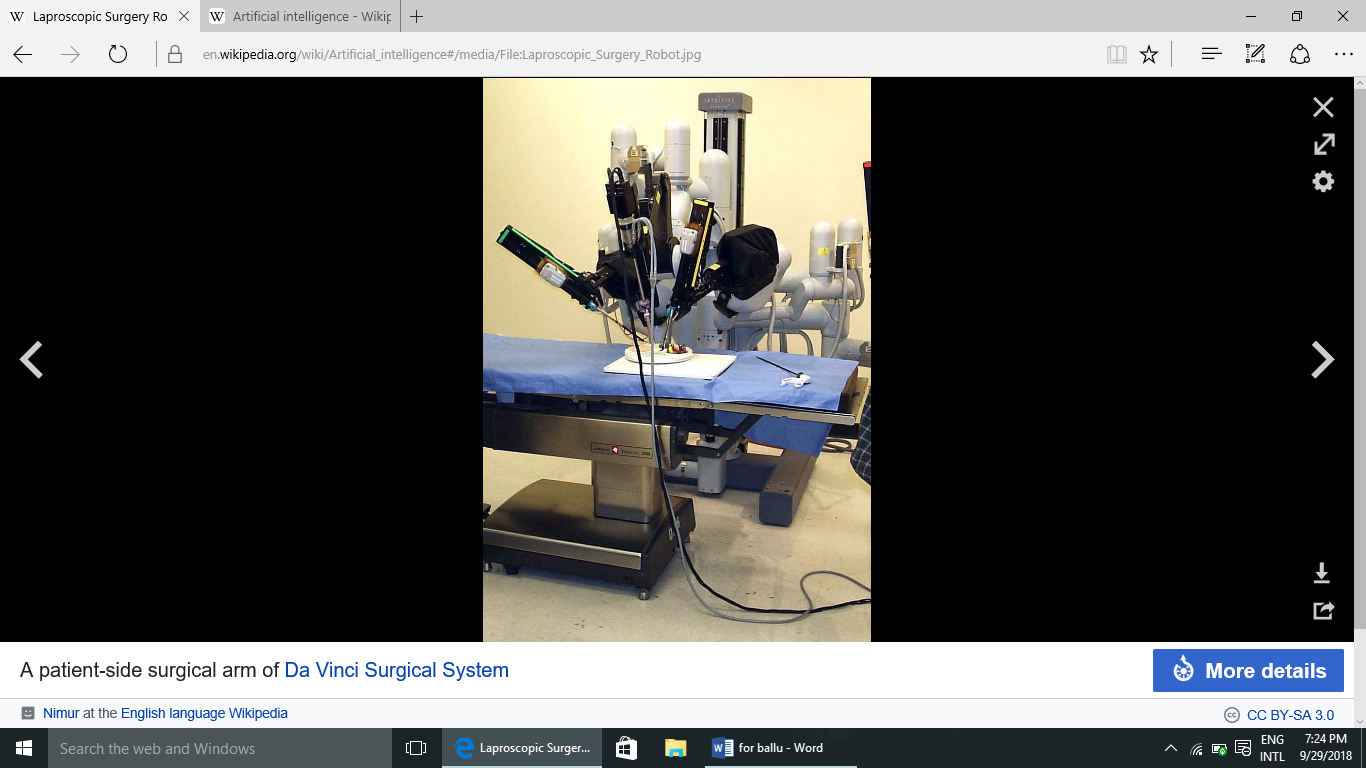
AI is relevant to any intellectual task. Modern artificial intelligence techniques are pervasive and are too numerous to list here. Frequently, when a technique reaches mainstream use, it is no longer considered artificial intelligence; this phenomenon is described as the AI effect.

High-profile examples of AI include autonomous vehicles (such as drones and self-driving cars), medical diagnosis, creating art (such as poetry), proving mathematical theorems, playing games (such as Chess or Go), search engines (such as Google search), online assistants (such as Siri), image recognition in photographs, spam filtering, prediction of judicial decisions and targeting online advertisements.

With social media sites overtaking TV as a source for news for young people and news organisations increasingly reliant on social media platforms for generating distribution,major publishers now use artificial intelligence (AI) technology to post stories more effectively and generate higher volumes of traffic.

### **3.3.4.1 HEALTHCARE**

*Main article: Artificial intelligence in healthcare*



A patient-side surgical arm of Da Vinci Surgical System

AI is being applied to the high cost problem of dosage issues—where findings suggested that AI could save $16 billion. In 2016, a ground breaking study in California found that a mathematical formula developed with the help of AI correctly determined the accurate dose of immunosuppressant drugs to give to organ patients.



X-ray of a hand, with automatic calculation of bone age by computer software

Artificial intelligence is breaking into the healthcare industry by assisting doctors. According to Bloomberg Technology, Microsoft has developed AI to help doctors find the right treatments for cancer. There is a great amount of research and drugs developed relating to cancer. In detail, there are more than 800 medicines and vaccines to treat cancer. This negatively affects the doctors, because there are too many options to choose from, making it more difficult to choose the right drugs for the patients. Microsoft is working on a project to develop a machine called "Hanover". Its goal is to memorize all the papers necessary to cancer and help predict which combinations of drugs will be most effective for each patient. One project that is being worked on at the moment is fighting myeloid leukemia, a fatal cancer where the treatment has not improved in decades. Another study was reported to have found that artificial intelligence was as good as trained doctors in identifying skin cancers.Another study is using artificial intelligence to try and monitor multiple high-risk patients, and this is done by asking each patient numerous questions based on data acquired from live doctor to patient interactions.

According to CNN, a recent study by surgeons at the Children's National Medical Center in Washington successfully demonstrated surgery with an autonomous robot. The team supervised the robot while it performed soft-tissue surgery, stitching together a pig's bowel during open surgery, and doing so better than a human surgeon, the team claimed. IBM has created its own artificial intelligence computer, the IBM Watson, which has beaten human intelligence (at some levels). Watson not only won at the game show *Jeopardy!* against former champions, but was declared a hero after successfully diagnosing a woman who was suffering from leukemia.

**4. PROJECT**

**4.1. INTRODUCTION TO PROJECT**

4.1.1. WHAT IS PATTERN RECOGNITION?

Pattern recognition is the automated recognition of patterns and regularities in data. Pattern recognition is closely related to artificial intelligence and machine learning, together with applications such as data mining and knowledge discovery in databases (KDD), and is often used interchangeably with these terms. However, these are distinguished: machine learning is one approach to pattern recognition, while other approaches include hand-crafted (not learned) rules or heuristics; and pattern recognition is one approach to artificial intelligence, while other approaches include symbolic artificial intelligence. A modern definition of pattern recognition is:

The field of pattern recognition is concerned with the automatic discovery of regularities in data through the use of computer algorithms and with the use of these regularities to take actions such as classifying the data into different categories.

This article focuses on machine learning approaches to pattern recognition. Pattern recognition systems are in many cases trained from labelled "training" data (supervised learning), but when no labelled data are available other algorithms can be used to discover previously unknown patterns (unsupervised learning). Machine learning is the common term for supervised learning methods and originates from artificial intelligence, whereas KDD and data mining have a larger focus on unsupervised methods and stronger connection to business use. Pattern recognition has its origins in engineering, and the term is popular in the context of computer vision: a leading computer vision conference is named Conference on Computer Vision and Pattern Recognition. In pattern recognition, there may be a higher interest to formalize, explain and visualize the pattern, while machine learning traditionally focuses on maximizing the recognition rates. Yet, all of these domains have evolved substantially from their roots in artificial intelligence, engineering and statistics, and they've become increasingly similar by integrating developments and ideas from each other.

Pattern recognition algorithms generally aim to provide a reasonable answer for all possible inputs and to perform "most likely" matching of the inputs, taking into account their statistical variation. This is opposed to pattern matching algorithms, which look for exact matches in the input with pre-existing patterns. A common example of a pattern-matching algorithm is regular expression matching, which looks for patterns of a given sort in textual data and is included in the search capabilities of many text editors and word processors. In contrast to pattern recognition, pattern matching is not generally a type of machine learning, although pattern-matching algorithms (especially with fairly general, carefully tailored patterns) can sometimes succeed in providing similar-quality output of the sort provided by pattern-recognition algorithms.

**4.2LIBRARY USED FOR PROJECT**

## 4.2.1 OPENCV

OpenCV was started at Intel in 1999 by **Gary Bradsky**, and the first release came out in 2000. **Vadim Pisarevsky** joined Gary Bradsky to manage Intel’s Russian software OpenCV team. In 2005, OpenCV was used on Stanley, the vehicle that won the 2005 DARPA Grand Challenge. Later, its active development continued under the support of Willow Garage with Gary Bradsky and Vadim Pisarevsky leading the project. OpenCV now supports a multitude of algorithms related to Computer Vision and Machine Learning and is expanding day by day.

OpenCV supports a wide variety of programming languages such as C++, Python, Java, etc., and is available on different platforms including Windows, Linux, OS X, Android, and iOS. Interfaces for high-speed GPU operations based on CUDA and OpenCL are also under active development.

OpenCV-Python is the Python API for OpenCV, combining the best qualities of the OpenCV C++ API and the Python language.

## 4.2.2 OPENCV-PYTHON

OpenCV-Python is a library of Python bindings designed to solve computer vision problems.

Python is a general purpose programming language started by **Guido van Rossum** that became very popular very quickly, mainly because of its simplicity and code readability. It enables the programmer to express ideas in fewer lines of code without reducing readability.

Compared to languages like C/C++, Python is slower. That said, Python can be easily extended with C/C++, which allows us to write computationally intensive code in C/C++ and create Python wrappers that can be used as Python modules. This gives us two advantages: first, the code is as fast as the original C/C++ code (since it is the actual C++ code working in background) and second, it easier to code in Python than C/C++. OpenCV-Python is a Python wrapper for the original OpenCV C++ implementation.

OpenCV-Python makes use of **Numpy**, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the OpenCV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

For import the library

1. **>>>importcv2**
2. **>>>print** cv2.\_\_version\_\_

## 4.2.3 GOALS

* Here, you will learn how to read an image, how to display it and how to save it back
* You will learn these functions :**cv2.imread()**, **cv2.imshow()** , **cv2.imwrite()**
* Optionally, you will learn how to display images with Matplotlib

**4.2.4 BASIC FUCTIONS**

### **4.2.4.1 Read an image**

Use the function **cv2.imread()** to read an image. The image should be in the working directory or a full path of image should be given.

Second argument is a flag which specifies the way image should be read.

* cv2.IMREAD\_COLOR : Loads a color image. Any transparency of image will be neglected. It is the default flag.
* cv2.IMREAD\_GRAYSCALE : Loads image in grayscale mode
* cv2.IMREAD\_UNCHANGED : Loads image as such including alpha channel

**Note:**Instead of these three flags, you can simply pass integers 1, 0 or -1 respectively.

See the code below:

importnumpyasnp

importcv2

# Load ancolor image in grayscale

img=cv2.imread('messi5.jpg',0)

Warning:Even if the image path is wrong, it won’t throw any error, but printimg will give you None

### **4.2.4.2 Display an image**

Use the function **cv2.imshow()** to display an image in a window. The window automatically fits to the image size.

First argument is a window name which is a string. second argument is our image. You can create as many windows as you wish, but with different window names.

cv2.imshow('image',img)

cv2.waitKey(0)

cv2.destroyAllWindows()

A screenshot of the window will look like this (in Fedora-Gnome machine):



**cv2.waitKey()** is a keyboard binding function. Its argument is the time in milliseconds. The function waits for specified milliseconds for any keyboard event. If you press any key in that time, the program continues. If **0** is passed, it waits indefinitely for a key stroke. It can also be set to detect specific key strokes like, if key a is pressed etc which we will discuss below.

Note

Besides binding keyboard events this function also processes many other GUI events, so you MUST use it to actually display the image.

**cv2.destroyAllWindows()** simply destroys all the windows we created. If you want to destroy any specific window, use the function **cv2.destroyWindow()** where you pass the exact window name as the argument.

Note:There is a special case where you can already create a window and load image to it later. In that case, you can specify whether window is resizable or not. It is done with the function **cv2.namedWindow()**. By default, the flag is cv2.WINDOW\_AUTOSIZE. But if you specify flag to be cv2.WINDOW\_NORMAL, you can resize window. It will be helpful when image is too large in dimension and adding track bar to windows.

See the code below:

cv2.namedWindow('image',cv2.WINDOW\_NORMAL)

cv2.imshow('image',img)

cv2.waitKey(0)

cv2.destroyAllWindows()

### **4.2.4.3 Write an image**

Use the function **cv2.imwrite()** to save an image.

First argument is the file name, second argument is the image you want to save.

cv2.imwrite('messigray.png',img)

This will save the image in PNG format in the working directory.

### **Sum it up**

Below program loads an image in grayscale, displays it, save the image if you press ‘s’ and exit, or simply exit without saving if you press ESC key.

importnumpyasnp

importcv2

img=cv2.imread('messi5.jpg',0)

cv2.imshow('image',img)

k=cv2.waitKey(0)

ifk==27:# wait for ESC key to exit

cv2.destroyAllWindows()

elifk==ord('s'):# wait for 's' key to save and exit

cv2.imwrite('messigray.png',img)

cv2.destroyAllWindows()

Warning:If you are using a 64-bit machine, you will have to modify k=cv2.waitKey(0) line as follows :

k=cv2.waitKey(0)&0xFF

## 4.2.4.4 Using Matplotlib

Matplotlib is a plotting library for Python which gives you wide variety of plotting methods. You will see them in coming articles. Here, you will learn how to display image with Matplotlib. You can zoom images, save it etc using Matplotlib.

importnumpyasnp

importcv2

frommatplotlibimportpyplotasplt

img=cv2.imread('messi5.jpg',0)

plt.imshow(img,cmap='gray',interpolation='bicubic')

plt.xticks([]),plt.yticks([])# to hide tick values on X and Y axis

plt.show()

A screen-shot of the window will look like this :



See also

Plenty of plotting options are available in Matplotlib. Please refer to Matplotlib docs for more details. Some, we will see on the way.

Warning

Color image loaded by OpenCV is in BGR mode. But Matplotlib displays in RGB mode. So color images will not be displayed correctly in Matplotlib if image is read with OpenCV. Please see the exercises for more details.

### **4.3 WHAT ARE THE STEP TO DETECT THE IMAGE?**

#### **4.3.1 How does it work?**

The code works simply as two parts. The aim of the first part is to train the script with possible images. Once the training is done, you can test your scanned image (i.e. credit card or insurance card).

### **4.3.2 Frameworks to be installed**

Python-2.7.x.  
numpy  
opencv

### **4.3.3 How to train?**

Open a terminal, go to the folder and run the command for train :

python digit\_recognizer\_training.py

### **4.3.4 How to test?**

Open a terminal, go to the folder and run the command

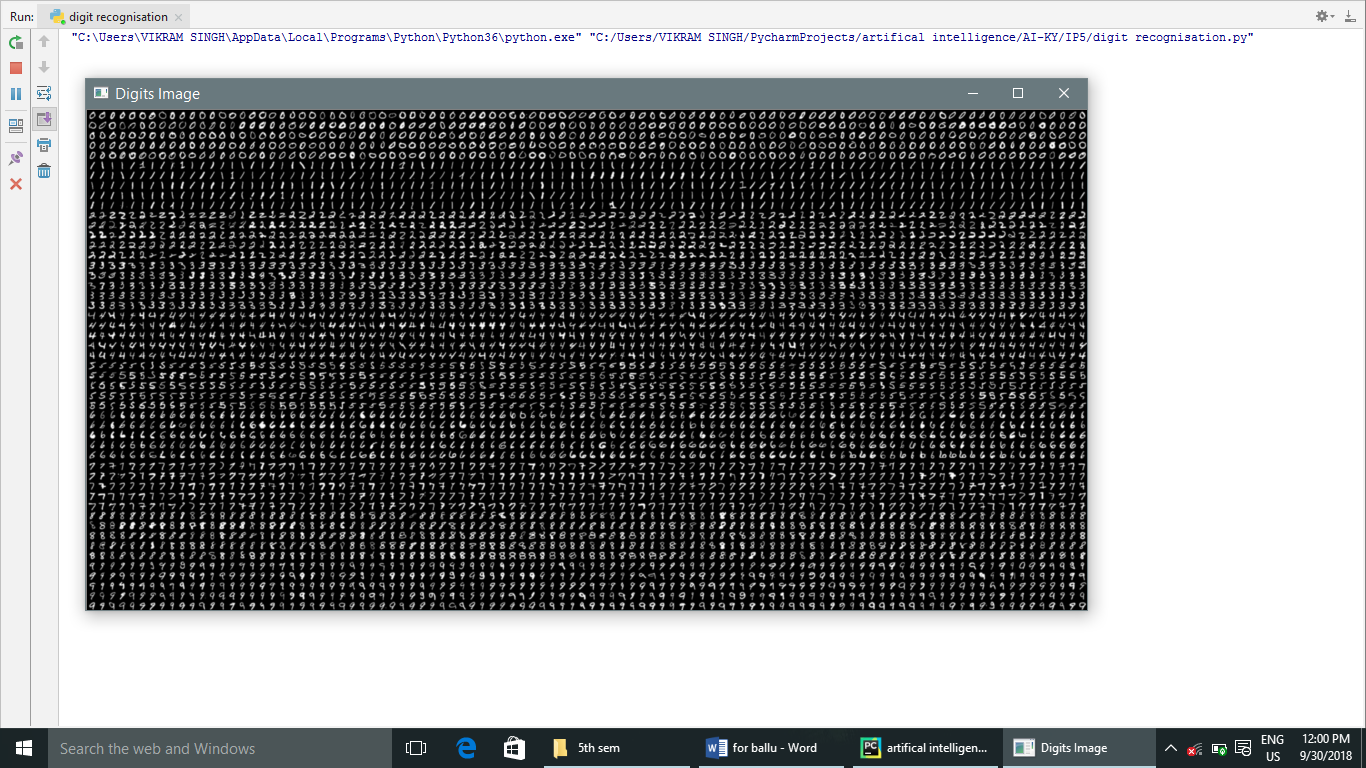
**5.SOURCE CODE**

**Import** numpy **as** np  
image = cv2.imread(**"digits.png"**)  
gray = cv2.cvtColor(image,cv2.COLOR\_BGR2GRAY)  
small = cv2.pyrDown(image)  
cv2.imshow(**"Digits Image”**, small)  
cv2.waitKey(0)  
cv2.destroyAllWindows()  
*#split the image to 5000 cells, each cell 20 x 20*cells = [np.hsplit(row,100) **for** row **in** np.vsplit(gray,50)]  
*#convert list data type to numpy array of shape (50,100,20,20)*x = np.array(cells)  
print(**"The shape of our cells array"**+str(x.shape))  
*#Split the full dataset into two segments*train = x[:,:70].reshape(-1,400).astype(np.float32) *#size = 3500x400*test = x[:,70:100].reshape(-1,400).astype(np.float32) *#size = 1500x400  
#create label for train and test data*k = [0,1,2,3,4,5,6,7,8,9]  
train\_labels = np.repeat(k,350)[:,np.newaxis]  
test\_labels = np.repeat(k,150)[:,np.newaxis]  
knn = cv2.ml.KNearest\_create()  
knn.train(train,cv2.ml.ROW\_SAMPLE,train\_labels)  
ret,results,neighbors,distance = knn.findNearest(test, k=3)  
matches = results == test\_labels  
correct =np.count\_nonzero(matches)  
accuracy = correct \* (100.0 / results.size)  
print(**"Accuracy is = %.2f"**% accuracy + **"%"**)  
**def** x\_cord\_contour(contour):  
**if** cv2.contourArea(contour)>10:  
 M = cv2.moments(contour)  
**return**

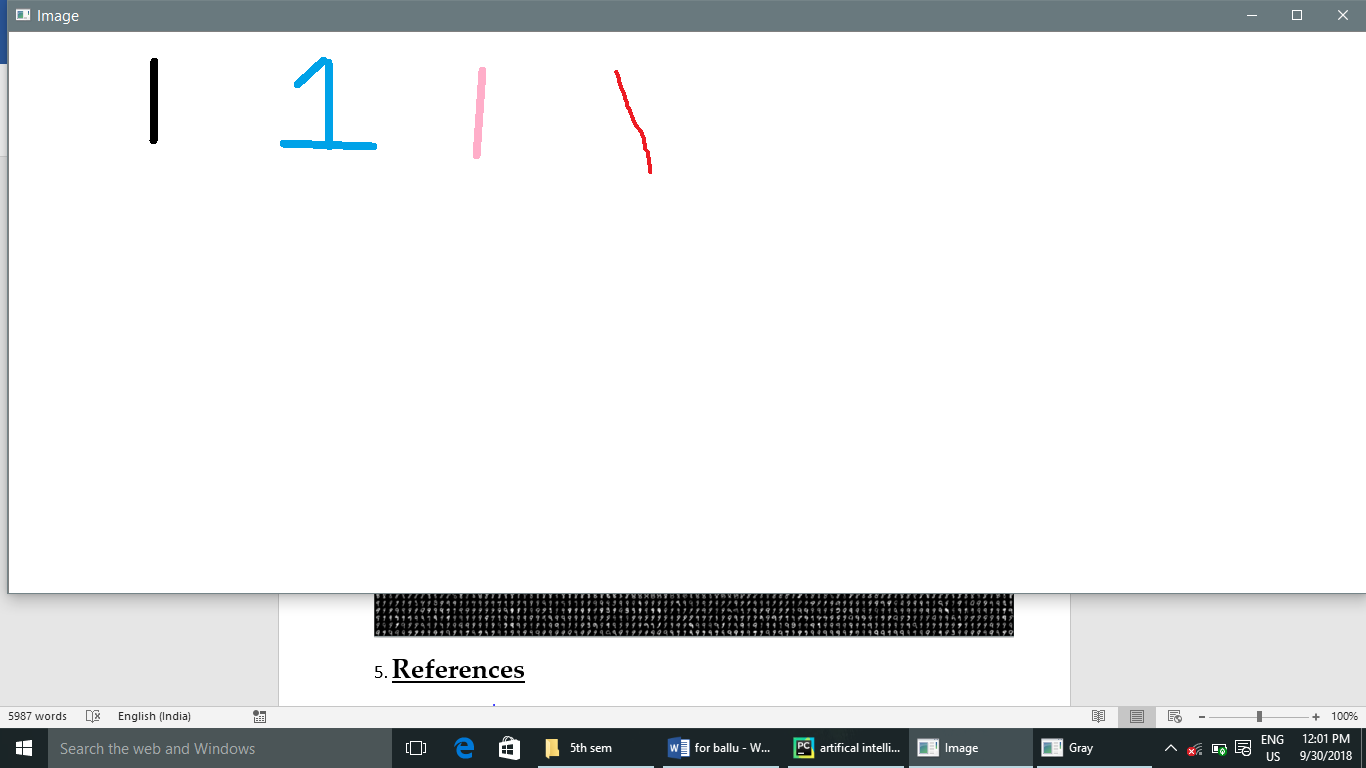
int(M[**'m10'**]/M[**'m00'**])  
**else**:  
**return** int(0)  
**def** makeSquare(not\_square):  
 BLACK = [0,0,0]  
img\_dim = not\_square.shape  
 height = img\_dim[0]  
 width = img\_dim[1]  
**if**(height == width):  
 square = not\_square  
**return** square  
**else**:  
doublesize = cv2.resize(not\_square,(2\*width, 2\*height), interpolation=cv2.INTER\_CUBIC)  
 height = height \* 2  
width = width \* 2  
**if**(height > width):  
 pad = int((height - width)/2)  
doublesize\_square = cv2.copyMakeBorder(doublesize,0,0,pad,pad,cv2.BORDER\_CONSTANT,value=BLACK)  
**else**:  
 pad = int((width - height)/2)  
doublesize\_square = cv2.copyMakeBorder(doublesize, pad,pad,0,0,cv2.BORDER\_CONSTANT, value=BLACK)  
doublesize\_square\_dim = doublesize\_square.shape  
**return** doublesize\_square  
**def** resize\_to\_pixel(dimensions, image):  
buffer\_pix =4  
dimensions = dimensions - buffer\_pix  
 squared = image  
 r = float(dimensions) / squared.shape[1]  
 dim = (dimensions,int(squared.shape[0]\*r))  
 resized = cv2.resize(image,dim,interpolation=cv2.INTER\_AREA)  
 img\_dim2 = resized.shape  
height\_r = img\_dim2[0]  
width\_r = img\_dim2[1]  
 BLACK = [0,0,0]  
**if**(height\_r>width\_r):  
 resized = cv2.copyMakeBorder(resized,0,0,0,1,cv2.BORDER\_CONSTANT,value=BLACK)  
**if** (height\_r>width\_r):  
 resized = cv2.copyMakeBorder(resized, 1, 0, 0, 0, cv2.BORDER\_CONSTANT, value=BLACK)  
 p = 2  
ReSizedImg = cv2.copyMakeBorder(resized, p, p, p, p, cv2.BORDER\_CONSTANT, value=BLACK)  
img\_dim = ReSizedImg.shape  
 height = img\_dim[0]  
 width = img\_dim[1]  
**return** ReSizedImg  
image2 = cv2.imread(**"ss.png"**)  
gray2 = cv2.cvtColor(image2,cv2.COLOR\_BGR2GRAY)  
cv2.imshow(**"Image"**,image2)  
cv2.imshow(**"Gray"**,gray2)  
cv2.waitKey(0)  
blurred = cv2.GaussianBlur(gray2,(5,5),0)  
cv2.imshow(**"Blurred"**,blurred)  
cv2.waitKey(0)  
edged = cv2.Canny(blurred, 30, 150)  
cv2.imshow(**"Edged"**,edged)  
cv2.waitKey(0)  
\_, contours, \_ = cv2.findContours(edged.copy(),cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)  
contours = sorted(contours, key = x\_cord\_contour, reverse = **False**)  
full\_number = []  
**for** c **in** contours:  
 (x,y,w,h) = cv2.boundingRect(c)  
 cv2.drawContours(image2, contours, -1,(0,255,0),3)  
 cv2.imshow(**"Contours"**,image2)  
**if** w >= 5 **and** h >= 25:  
roi = blurred[y:y + h, x:x +w]  
 ret, roi = cv2.threshold(roi,127,255,cv2.THRESH\_BINARY\_INV)  
 squared = makeSquare(roi)  
 final = resize\_to\_pixel(20, squared)  
 cv2.imshow(**"Final"**,final)  
final\_array = final.reshape((1,400))  
final\_array = final\_array.astype(np.float32)  
 ret, result, neighbors, dist= knn.findNearest(final\_array, k=1)  
 number = str(int(float(result[0])))  
full\_number.append(number)  
 cv2.rectangle(image2,(x,y),(x+w,y+h),(0,0,255),2)  
 cv2.putText(image2,number,(x,y+155),  
 cv2.FONT\_HERSHEY\_COMPLEX,2,(255,0,0),2)  
 cv2.imshow(**"IMage2"**,image2)  
 cv2.waitKey(0)  
cv2.destroyAllWindows()

**6. OUTPUTS**

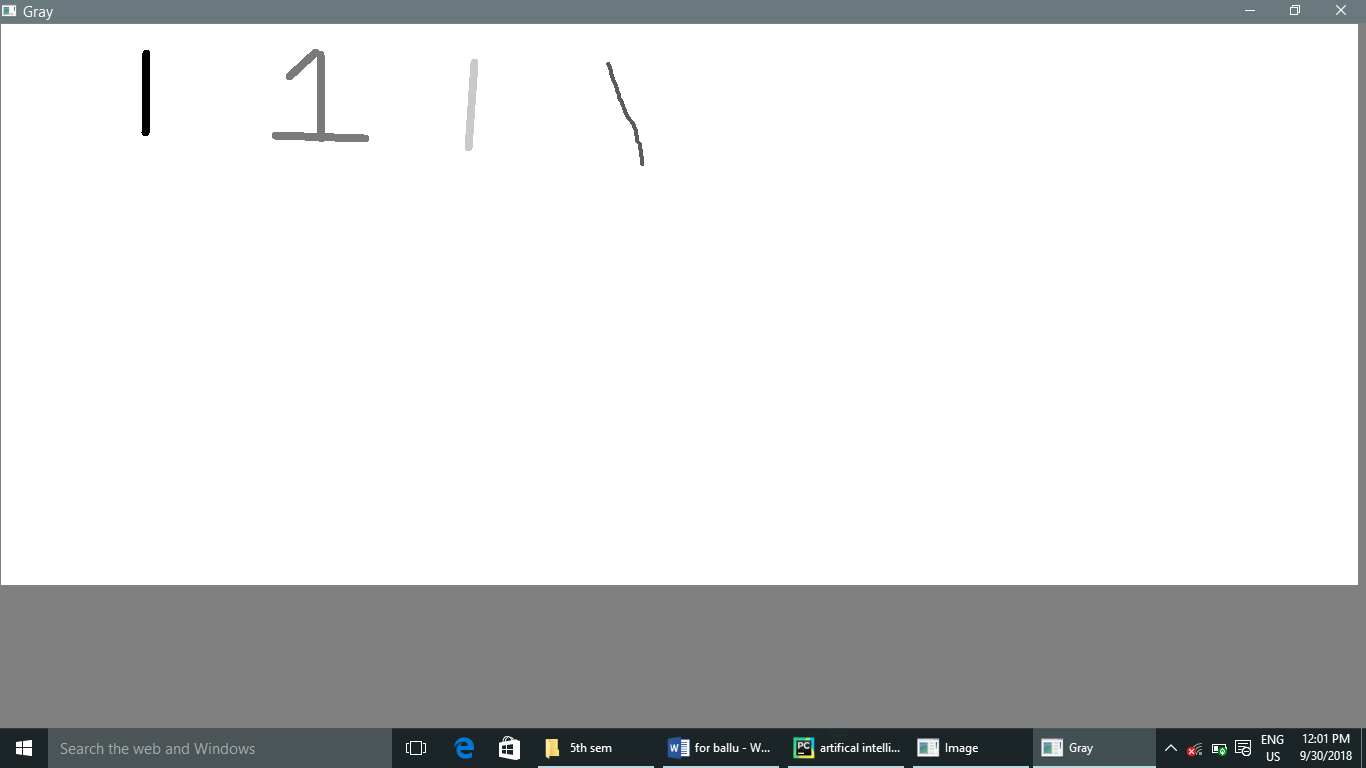
**DATA SET :**



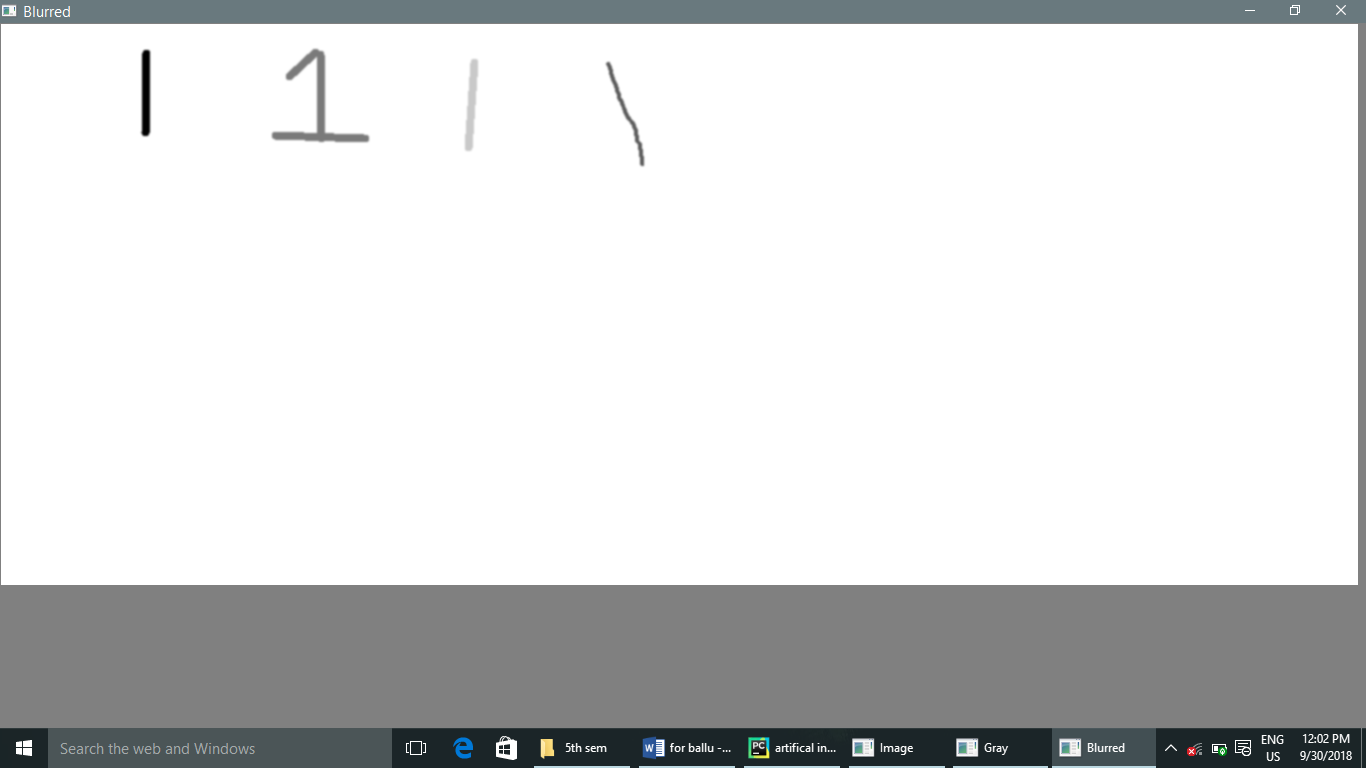
**TRAINER KIT:**



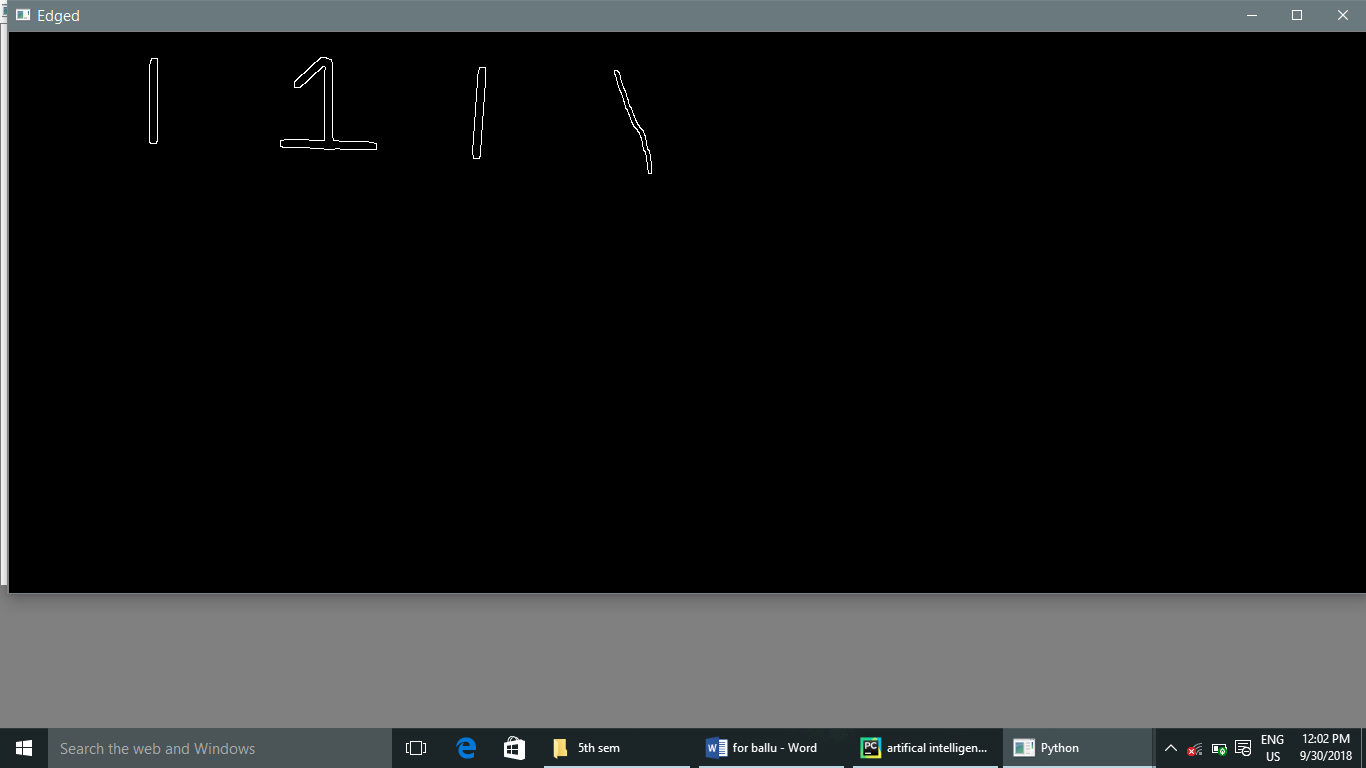
Step 1- conver the image into the gray scale.



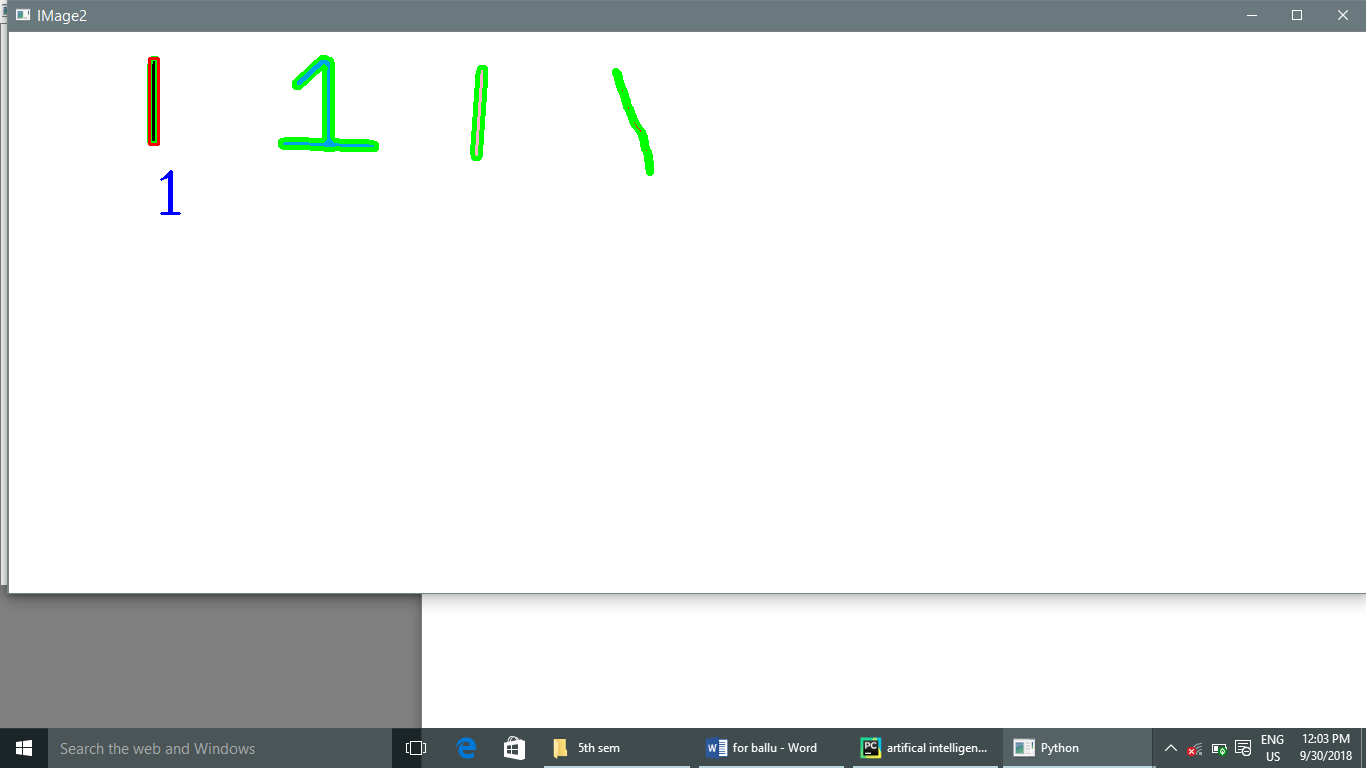
Step 2- convert the gray scale image to blur image using any method.

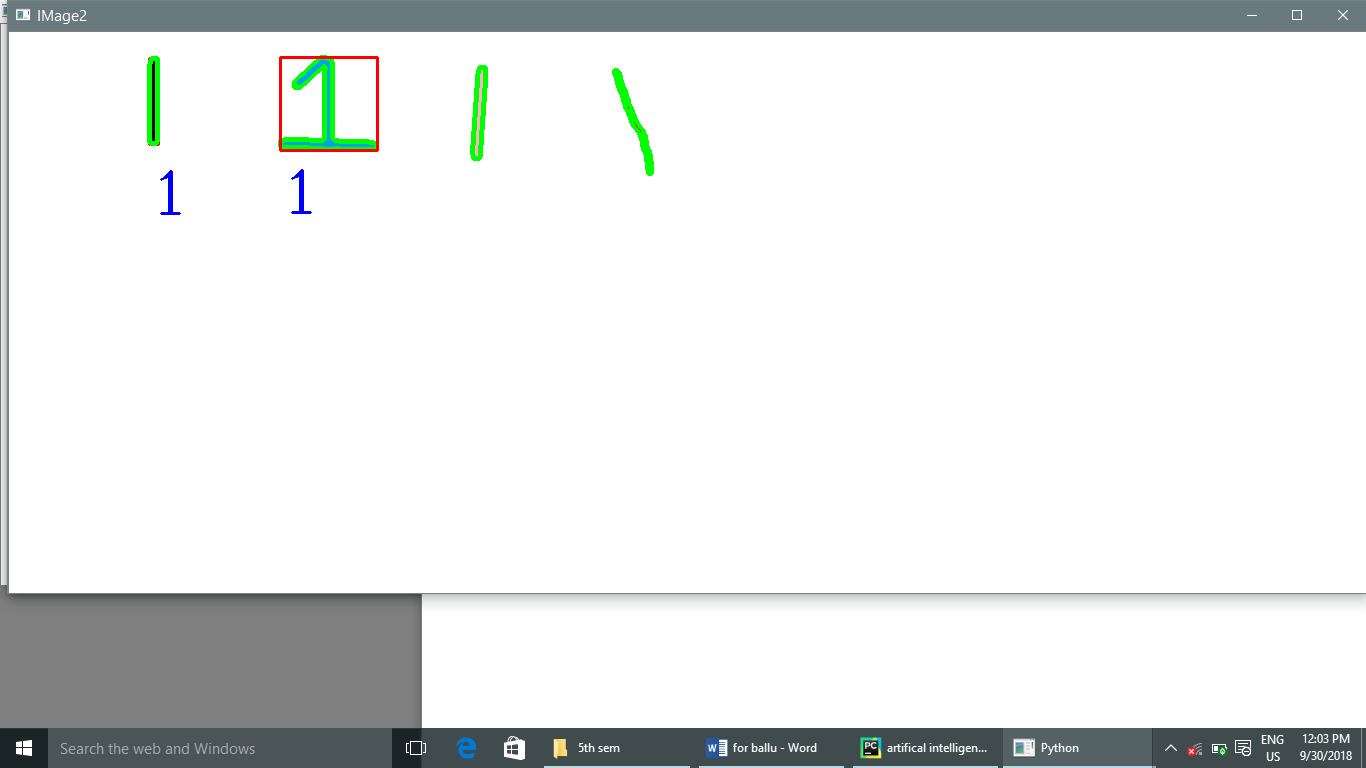


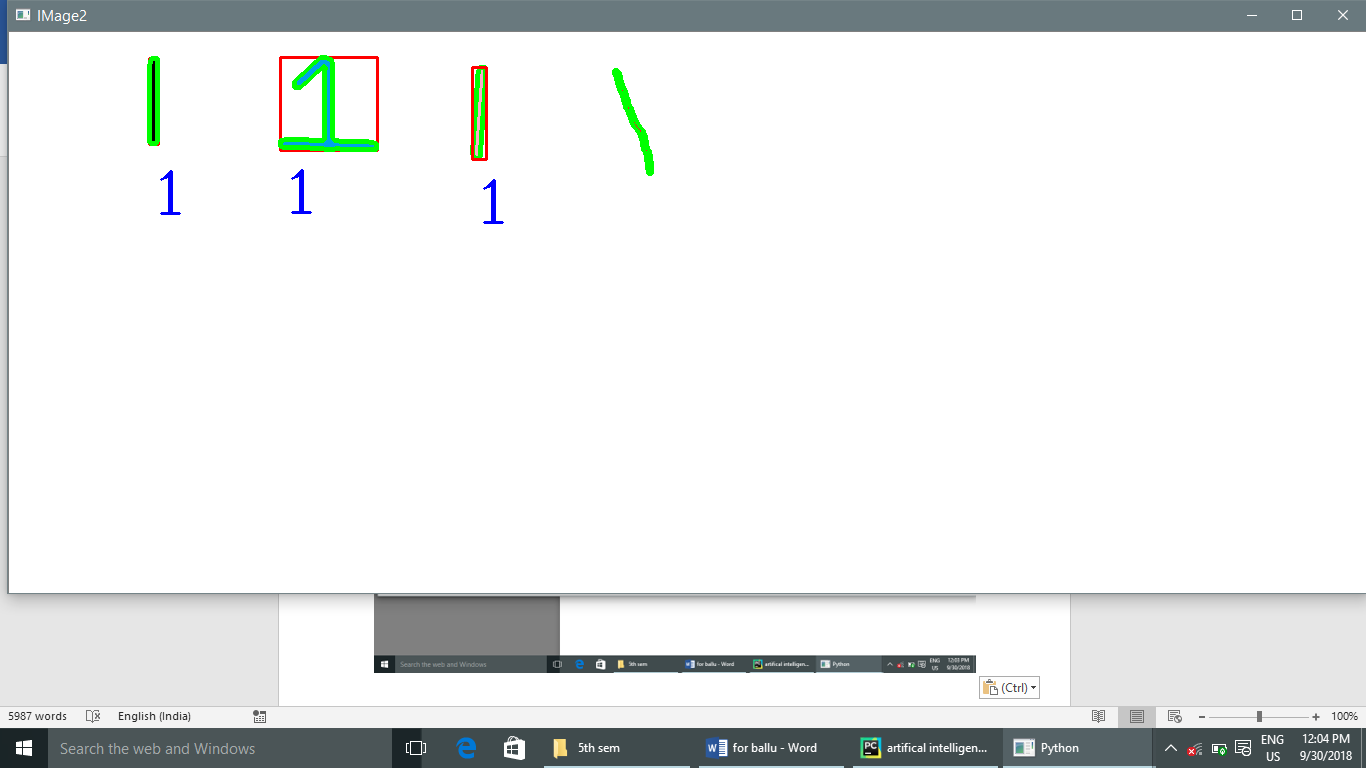
Step 3- detect the edges from the blurred image.

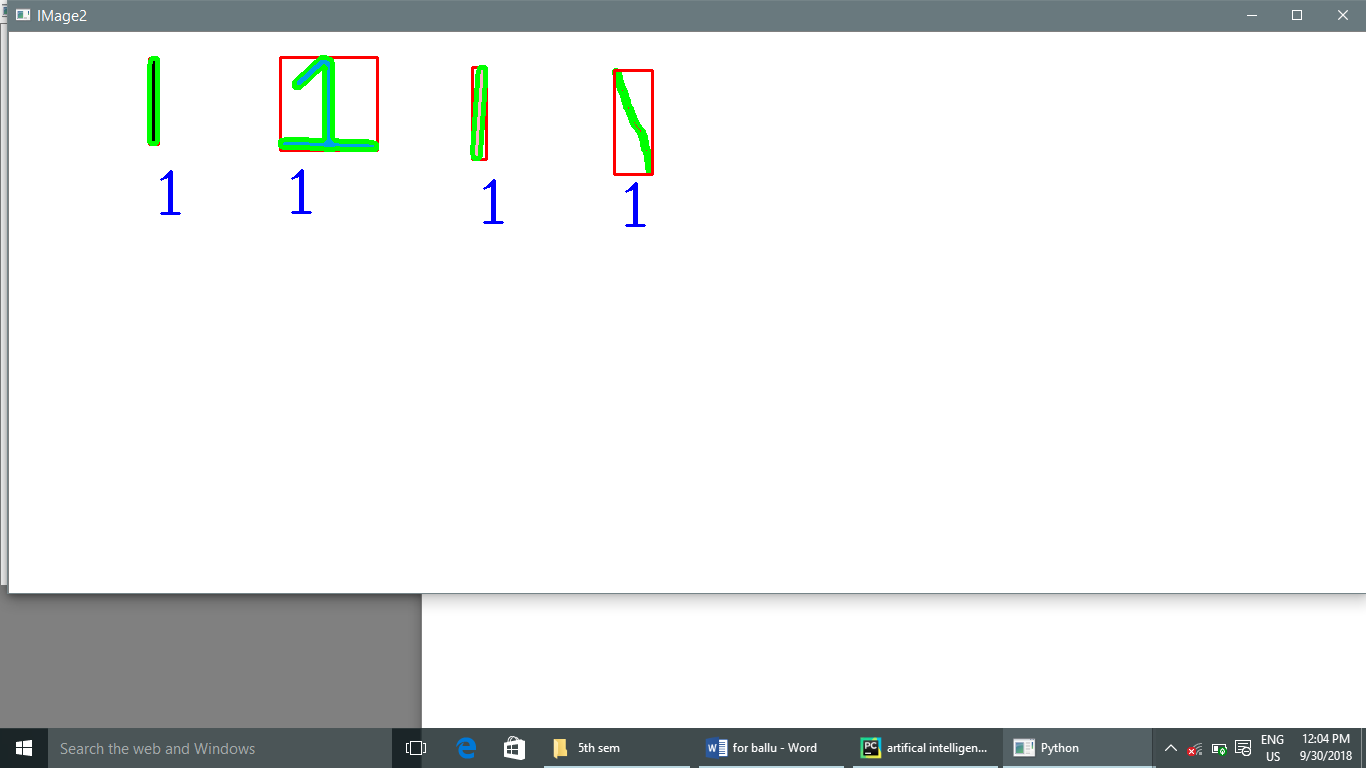


Step 4- generate the contour in the image for detection of number from data set.









**7. REFERENCES**

* www.google.com
* www.opencv-python tutorial .com
* www.python.org
* www.wikkipedia.org
* www.opencv.org
* Complete Reference to python and Artificial intelligence.
* HPE Study Material.